ATARI

ST COMPUTER

Die Fachzeitschrift für den ATARI-ST Anwender.

Oktober '87

10

DM 6,–
Ös 53,–/Sfr. 6,–

PUZZLE
Video Digitizer im Selbstbau

ALICE IM WUNDERLAND
Ein Pascal Interpreter

INS NETZ GEGANGEN
Ein Netzwerk für den ST

TRUE BASIC
Das wahre Basic?

Neue Projekte

Liebe Leserin, lieber Leser,

wie im Modegeschäft, so präsentieren auch die Computerfirmen jedes Jahr im Frühjahr und im Herbst ihre neuste „Kollektion". Orte hierfür sind die großen Messen, ist es im März die „CeBit" in Hannover, so ist es im Oktober zur Abwechslung die „Orgatechnik" in Köln und die „Systems" in München. In Amerika wären die „CES" und die „Comdex" zu nennen. In diesem Jahr ist wieder München, vom 19. bis 23. Oktober mit der 10. Fachmesse für Computer und Kommunikation, der Systems 87, Schauplatz für viele Computerinteressenten. Obwohl ATARI diesmal keine weltbewegenden Neuigkeiten angekündigt hat, lohnt sich ein Besuch mit Sicherheit. Unser Stand ist im Informationszentrum Halle 21 (Stand-Nr. C15/C17). Dort können Sie sich über unsere Produkte informieren, Fragen stellen und Wünsche äußern. Das neuste Projekt, unser Videodigitizer, wird selbstverständlich auch vorgeführt.

An dieser Stelle möchte ich erwähnen, daß wir einem häufig geäußerten Wunsch unserer Leser, mehr Hardware zu bringen, in der Zukunft größere Aufmerksamkeit widmen werden. Das erste Hardware-Projekt, unser Easytizer (siehe Bild — schwarzweiß Digitalisierung mit 640 x 400 Bildpunkten Auflösung), beginnt bereits in diesem Heft. Um lesernahe Projekte entwickeln zu können, bitte ich Sie, lieber Leser, uns Ihre Wünsche, Probleme oder Ideen, die sich als Projekt veröffentlichen lassen, mitzuteilen. Dazu können Sie die im Heft befindliche Postkarte an die Redaktion verwenden. Natürlich sind auch Ihre fertigen Schaltungsideen bei uns jederzeit willkommen.

Damit sich das Schreiben auch für Sie lohnt, verlosen wir unter den Einsendungen „Hardware-Projekt", gleichgültig ob es sich dabei um ein Angebot oder eine Idee bzw. Wunsch handelt, einen fertig aufgebauten Easytizer und neun weitere Überraschungspreise. Einsendeschluß ist der 31.10.87 (Datum des Poststempels). Der Rechtsweg ist ausgeschlossen. Viel Glück!

Ihr

Uwe Bärtels

Allgemeines

Software

Bild 2: TIGER.IMG ohne 1st Patch

Bild 3: TIGER.IMG mit 1st Patch

1st Patch

24 Nadeln machen Druck

Bis vor kurzem erschienen Wordplus Grafiken nur mager auf einem 24-Nadeldrucker. 1st Patch ändert dies, indem er alle 8-Nadel Daten auf 24-Nadeln umrechnet. Das Resultat: ein satter Druck.

Salve Megamaxus

Interrupts in Megamax C

Ein Programm läßt sich nicht gerne unterbrechen. Die universelle Interruptroutine jedoch klinkt sich 100 mal pro Sekunde ein, ohne Rücksicht auf das Hauptprogramm zu nehmen.

ALICE aus dem Wunderland

Ein Pascal Interpreter

Nein, kein Druckfehler: ALICA-Pascal ist der erste Pascal Interpreter für den ST. Das bietet völlig neue Möglichkeiten für die Programmierwelt. Ideal anwendbar ist er für Anfänger.

Hardware

Puzzle

Ein Videodigitizer im Selbstbau

Unsere Hardwareentwickler haben sich entschlossen, dem ST ein elektronisches Auge zu konstruieren. Diese anspruchsvolle Schaltung stellt den Anfang für viele noch folgende Hardwareprojekte dar. Viel Spaß beim Nachbauen.

Pascal ruft zur Ordnung

Algorithmen und Datenstrukturen in Pascal

Die berühmteste Sprache von Niklaus Wirth bietet die besten Möglichkeiten zur Entwicklung von Algorithmen. Diese Serie soll die verschiedensten Datenstrukturen und deren Programmiertechnik anschaulich machen.

Fünf atü

Kräftig aufgeblasener Diskmonitor

In diesem Teil wollen wir die Theorie in die Praxis umsetzen. ‚Extended MINIMON' enthält viele Programmiertricks, die man zum Floppy benötigt.

Grundlagen

Kurse

Aktuelles

PINBOARD

30 OMIKRON.Basic-Compiler zu gewinnen

Wir suchen Tips & Tricks in OMIKRON.Basic. Da bekanntlich gerade kleinere Routinen für die Anwender von Interesse sind, möchten wir hier unsere Leser aufrufen, falls sie solche zur Hand haben, uns diese zu schikken. Dabei ist es egal ob es sich um eine eigenständiges Programm, eine Prozedur oder Tips & Tricks anderer Art handelt. Wichtig ist nur, daß auch andere Leser daraus einen Nutzen ziehen können.

Gestiftet von Omikron-Software

Als Belohnung erhält jeder Einsender, dessen Tip veröffentlicht wird, einen OMIKRON.Compiler. Max. 30 Stück sind zu vergeben.

Der Rechtsweg ist ausgeschlossen. Die Teilnahme von Redaktions- und Verlagsangehörigen ist nicht zugelassen.

Einsendeschluß ist der 11. 11. 1987

'MERLIN'-Computer GmbH
'Tips & Tricks OMIKRON.Basic'
Industriestr. 26
D-6236 Eschborn

5 LATTICE C zu gewinnen

LATTICE C gibt es nun in der neuen Version 3.04 – und Sie können einen davon gewinnen. Schicken Sie uns dazu ein kleines Programm, das in C geschrieben wurde ein. Das Programm (Progrämmchen) sollte eine interessante Funktion erfüllen, die anderen Programmierern von Nutzen ist.

Gestiftet von GFA-Systemtechnik

Die Auswahl der Gewinner wird von einer redaktionsinternen Jury vorgenommen.

Der Rechtsweg ist ausgeschlossen. Die Teilnahme von Redaktions- und Verlagsangehörigen ist nicht zugelassen.

Einsendeschluß: 11. 11. 1987

'MERLIN'-Computer GmbH
'LATTICE C'
Industriestr. 26
D-6236 Eschborn

5 GFA-Publisher zu gewinnen

Wer möchte gerne einen Desktop Publisher gewinnen ? Hier bieten wir die Chance. Senden Sie dazu einfach eine kleine nützliche Routine in GFA Basic ein. Die 5 besten Einsendungen erhalten je ein GFA-Publisher. Die Auswahl der Gewinner wird von einer redaktionsinternen Jury vorgenommen.

Gestiftet von G. Knupe GmbH & Co. KG

Der Rechtsweg ist ausgeschlossen.

Angehörigen der Redaktion oder des Verlags ist die Teilnahme untersagt.

Einsendeschluß: 11. 11. 1987

Merlin-Computer GmbH
'GFA-Publisher'
Industriestr. 26
D-6236 Eschborn

Erfahrungen gesucht

Anwenderprogramme bieten meist viele Möglichkeiten, aber leider auch einige Schwierigkeiten. Oft liegen manche Fähigkeiten auch im Verborgenen. Deshalb möchten wir Sie bitten, uns Ihre Erfahrungen mit Anwenderprogrammen (z.B. Wordplus, Signum, Adimens, VIP, Logistiks, etc.) oder zu div. Utilities zu schildern, um anderen Lesern wertvolle Tips geben zu können.

Wir werden diese im Rahmen der Möglichkeit in der ST-Computer veröffentlichen. Natürlich werden diese Tips honoriert.

'MERLIN'-Computer GmbH
ST-Computer Redaktion
'Anwenderprogramme'
Industriestr. 26
D-6236 Eschborn

BILLIG-SPIELE FÜR AMIGA und ST MÜSSEN NICHT »BILLIG« SEIN!

MÜLL

CITY DEFENCE

von Uwe Kühner

Eine schier endlose Zahl tödlicher Laserstrahlen aus dem Weltall kann nur gestoppt werden, wenn Sie zum richtigen Zeitpunkt Ihre Bodenraketen abschiessen, bevor die Städte zerstört werden. 1/2 Spieler; Maussteuerung. **Lieferbar für AMIGA**

GOKART RACING

von Anco

Ein exzellentes Autorennen, bei dem ein echter Champion gefordert wird. Bei unterschiedlichsten Witterungsbedingungen und Strecken müssen Sie vorher Reifen und Übersetzung wählen. 8 Kurse, 1/2 Spieler (gleichzeitig); Joysticksteuerung.
Lieferbar für AMIGA und ST

EMERALD MINE

von Klaus Heinz und Volker Wertich

Ein Riesen-Arcade-Adventure mit über 100 Leveln und 2-Spieler-Option (gleichzeitig im Teamwork!); Joysticksteuerung. *"Die flotte Grafik mit witzigen Monster-Sprites und vor allem der Sound gefallen mir sehr gut: Die fetzige Titelmusik und die digitalisierten Sound-Effekte sind exzellent. Das Verblüffendste an diesem tollen Spiel ist aber der sehr niedrige Preis: Wer hier nicht zuschlägt, ist selber schuld." (HAPPY COMPUTER)*
Lieferbar für AMIGA und ST

STRIP POKER

von Artworx

Verbringen Sie einen schönen Abend mit einer Runde Strip Poker. Mit Susi und Melissa stehen Ihnen 2 bildschöne, aber spielstarke Partnerinnen zur Verfügung. Bequeme Spielsteuerung mit der Maus. **Lieferbar für AMIGA und ST**

FORTRESS UNDERGROUND

von Andreas von Lepel

In einer gigantischen unterirdischen Höhle (640 Bildschirme groß) müssen Sie ein feindliches Kraftwerk aufspüren und eliminieren. Aber der Weg dorthin ist lang und gefährlich. Joysticksteuerung.
Lieferbar für AMIGA

DEMOLITION – Lieferbar für AMIGA

CRUNCHER FACTORY – Lieferbar für AMIGA

Flip Flop – Lieferbar für AMIGA/ST

Lieferbar für AMIGA

Lieferbar für ST

PHALANX – Lieferbar für AMIGA

SPACE BATTLE – Lieferbar für AMIGA

Space Pilot – Lieferbar für ST

TERRA NOVA – Lieferbar für ST

WILLY THE KID – Lieferbar für AMIGA/ST

SENSATIONELL

je Spiel nur DM

29.95

unverb. Preisempfehlung

KINGSOFT-Spiele gibt's überall da, wo es gute Software gibt.

SPITZEN-SOFTWARE ...natürlich von

KINGSOFT

MADE IN GERMANY

WANTED

PRO GRAMMIERER

Bei uns gibt's jeden Monat Neuerscheinungen für Ihren AMIGA und ST - am besten kostenlos den aktuellen Katalog anfordern!

Alle Spiele selbstverständlich mit ausführlicher deutscher Anleitung, Super-Grafik und fantastischen Digisound-Effekten.

F. Schäfer · Schnackebusch 4 · 5106 Roetgen
☎ 02408/5119 (nicht aufgeben!) Fax 02408-5213

DER DRUCKER

Stellen Sie sich einmal folgende Situation vor. Sie verfügen über einen leistungsfähigen Computer. Des weiteren sind Sie im Besitz einer vorbildlich und hundertprozentig nach Ihren Wünschen funktionierenden Software. Die Ergebnisse auf dem Bildschirm sehen ebenfalls noch erstklassig aus. Wenn es dann aber darum geht, Druck zu machen, fehlen Ihnen die Worte. Denn Ihr Drucker kann leistungsmäßig nicht mithalten und bietet eine blasse Vorstellung. Das kommt Ihnen bekannt vor? Dann gibt es nur eines. Schwenken Sie um und stellen Sie Ihrem Computer einen gleichwertigen Partner an die Seite. Zum Beispiel den NEC Pinwriter P6 Color. 24 Nadeln garantieren perfekte Druckergebnisse in schwarzweiß und Farbe.

Pinwriter P6 Color

NEC Pinwriter P6 Color.
Ein vorbildlicher Partner für Ihren Computer.

Grafiken, Text und EDV-Listen hinterlassen jederzeit den besten Eindruck. Und selbst komplexe Desktop Publishing- oder CAD-Aufgaben sind für einen Drucker wie den NEC Pinwriter P6 Color eine leichte Übung. Vom günstigen Preis einmal ganz zu schweigen.

NEC Pinwriter P6 Color
- **24-Nadel-Drucktechnologie**
- **Druckgeschwindigkeit: max. 216 Zeichen/Sek.**
- **Schreibbreite: 80 Zeichen/Zeile**
- **Auflösung: 360 x 360 Punkte/Zoll**
- **Betriebsgeräusch: 53 dBA (Quiet Mode)**

DER TREIBER

Sie wissen es nur zu gut: Eine Kette ist so stark wie ihr schwächstes Glied. Viele Computer/Software/Drucker-Konfigurationen haben ebenfalls eine klare Schwachstelle: den Druckertreiber. Mit einem qualitativ minderwertigen Treiber werden die dem Drucker seitens Computer und Software übermittelten Informationen nur unvollkommen umgesetzt und zu Papier gebracht. Die Ergebnisse sind dann naturgemäß enttäuschend. NEC läßt es nicht dazu kommen. Denn die eigenentwickelten bzw. in enger Zusammenarbeit mit den wichtigen Softwarehäusern erstellten Druckertreiber von NEC garantieren beste Ergebnisse. Nur mit einem perfekten Druckertreiber ist die Software in der Lage, alle Leistungen des Druckers vollkommen auszunutzen. Alle NEC Druckertreiber setzen die Computerdaten für den Drucker hundertprozentig verständlich um. Welche Programme von den NEC Pinwritern erstklassig unterstützt werden, läßt sich leicht ermitteln. Denn zu diesem Zweck gibt es den NEC Software Report.

- **Ein qualitativ minderwertiger Druckertreiber beeinträchtigt den Kommunikationsfluß zwischen Computer und Drucker.**

Textprogramme

Programm-Name	P560 P565	P560XL P565XL	P660 P665	CP660 CP665	P760 P765	CP760 CP765
AGTEXT	DOS	DOS	DOS	DOS	DOS	DOS
Atari 1st Word V 106	NEC	KF	NEC	KF	NEC	KF
Easy 10	ORG	KF	ORG	KF	ORG	KF
Easywriter 2	KOM	KOM	KOM	KOM	KOM	KOM
Signum	ORG	KF	ORG	KF	ORG	KF
Euroscript V 2	ORG	ORG	ORG	ORG	ORG	ORG
Gem Write	NEC	NEC	NEC	NEC	NEC	NEC
Info-Text	ORG	KF	ORG	KF	ORG	KF
MS Window Write	KOM	KOM	KOM	KOM	KOM	KOM
MS Word V 2	ORG	ORG	ORG	ORG	ORG	ORG
MS Word V 3	ORG	ORG	ORG	ORG	ORG	ORG
Multimate 3 31	ORG	ORG	ORG	ORG	ORG	ORG
Papyrus	NEC	KF	NEC	KF	NEC	KF
PC Text 3	NEC	NEC	NEC	NEC	NEC	NEC
PC-Write	ORG	KF	ORG	KF	ORG	KF
Profi-Text	ORG	ORG	ORG	ORG	ORG	ORG
Prosa	ORG	KF	ORG	KF	ORG	KF
Rechentext	ORG	ORG	ORG	ORG	ORG	ORG
Samna Word 3	ORG	ORG	ORG	ORG	ORG	ORG
Science Text V 657	ORG	KF	ORG	KF	ORG	KF
ST Textomat	ORG	ORG	ORG	ORG	ORG	ORG
Tex Ass Window +	NEC	NEC	NEC	NEC	NEC	NEC
Textomat PC	ORG	ORG	ORG	ORG	ORG	ORG
Volkswriter	ORG	ORG	ORG	ORG	ORG	ORG
Wordperfect	NEC	NEC	NEC	NEC	NEC	NEC
Wordstar 2000 V 101	ORG	KF	ORG	KF	ORG	KF
Wordstar 3 4	NEC	NEC	NEC	NEC	NEC	NEC
Wordstar 3 45	ORG	ORG	ORG	ORG	ORG	ORG

Auszug aus NEC Software Report (2/87)

DIE QUALITÄT

Dies dürfte selbst dem absoluten Drucker-Neuling sofort einleuchten. 24-Nadel-Drucker sind qualitativ den 9-Nadel-Druckern überlegen, und daß Drucker mit 24-Nadel-Technologie bessere Druckergebnisse liefern, ist folglich auch keine Frage. Hier landet man zwangsläufig bei NEC.

- **Nur NEC Pinwriter, die mit dem neuen Benutzerhandbuch in Deutsch und Original-NEC-Seriennummer ausgeliefert werden, sind vollkommen in das NEC-Service-paket integriert. Hierzu zählt u. a. 12-Monate-Garantie, Treiber-Software, telefonische Hotline und vieles mehr.**

Original NEC Bedienungshandbuch (Ringbuch, 312 Seiten)

Die 24 Nadeln der NEC Pinwriter erzeugen ein optimales Druckbild, gleichgültig ob bei Text-, Grafik-, CAD- oder Desktop Publishing-Anwendungen. Sie wissen es ja: Der Drucker ist quasi die Visitenkarte Ihres Computers. Und nur Spitzengeräte bringen die Leistung Ihres Computers in vollem Umfang zum Ausdruck.

Der neue Standard:
Betrachtet man einmal die Ausdruckqualität der NEC Pinwriter und vergleicht diese mit der anderer Geräte in der 24-Nadel-Klasse, wird deutlich, warum in Testberichten der Computer-Fachpresse die Leistung und Qualität der NEC Pinwriter immer häufiger als Standard angegeben wird. Diesem Urteil der Fachpresse ist nichts mehr hinzuzufügen.

NEC PINWRITER UND IHR PERSONAL COMPUTER

NEC Pinwriter sind die vorbildliche Ergänzung zu Ihrem Personal Computer. Dabei spielt es keine Rolle, ob Sie sich vornehmlich mit Text, CAD oder Grafik beschäftigen. Durch die perfekte Anpassung bilden Ihr Personal Computer und der NEC Pinwriter eine leistungsstarke Einheit. Und schaut man sich im NEC Software Report einmal die umfangreichen Software-Lösungen an, die die 24-Nadel-Technologie der NEC Drucker bis ins Kleinste ausnutzen, wird deutlich, warum immer mehr Personal Computer-User sich für einen NEC Pinwriter entscheiden.

NEC Pinwriter und Ihr Personal Computer: Ein Gespann, das es in sich hat.

Pictures by PC, 360 x 360 dpi. Ausschnittvergrößerung

GEM Graph

Jede dieser Schriftarten gibt doppelter oder doppelter Höhe, Fettschrift, Tiefstellung. Mit entsprechende SCHREIBEN IN VERSALIEN

Auszug MS Word 3,0

NEC PINWRITER UND IHR ATARI ST

Atari ST-Anwender sind von der Grafikleistung der NEC Pinwriter tief beeindruckt. Kein Wunder, bringen die NEC Pinwriter doch die Bildschirminformationen durch die NEC Grafik-Routine erstklassig zu Papier. Atari ST-User wollen natürlich einen perfekten Drucker, der, ebenso wie Ihr Computer, jederzeit einfach zu bedienen ist. Wenn auf dem Atari ST Programme wie 1. Word oder Signum! im Textbereich, Profi Painter für den „Grafiker" bzw. STAD, GFA-Draft oder CAMPUS für CAD-Anwendungen laufen, weiß man die Qualitäten der NEC Pinwriter zu schätzen.

Atari Hardcopy

Signum!

Einfach professionell: Atari ST und NEC Pinwriter.

Profi Painter

NEC PINWRITER UND IHR COMMODORE AMIGA

Die Fähigkeiten des Commodore Amiga liegen unbestritten im Anwendungsgebiet Grafik. Aber was auf dem Bildschirm leuchtend und kristallklar auftaucht, soll natürlich auch auf Papier perfekt aussehen. Und so mancher Drucker verliert hier plötzlich alle Farbe. Deshalb ist allen Commodore Amiga-Anwendern der NEC Pinwriter P6 Color zu empfehlen. Er gibt Grafiken nuancenreich und detailgenau wieder. Bei einer Auflösung von 360 x 360 Punkte/Zoll nicht weiter verwunderlich. Da die 24 Nadeln einzeln adressierbar sind, lassen sich selbst hochaufgelöste Bilder perfekt ausdrucken, wie die drei Printouts mit Deluxe Paint beweisen.

Ideales Paar: Ihr Commodore Amiga und der NEC Pinwriter P6 Color.

NEC

NEC Deutschland GmbH

NEC Deutschland GmbH 1000 Berlin 30 Tel.: 030/8826320
CHS 2000 Hamburg 1 Tel.: 040/23 22 23
L + S 3000 Hannover 81 Tel.: 0511/8 42 44-0
SYSDAT 5000 Koln 40 Tel.: 0221/4 89 05-0
S.E.H. 6455 Erlensee Tel.: 06183/8 30
Magirus 7022 Leinfelden-Echterdingen Tel.: 0711/7 59 04-0
SCHWIND 8033 Martinsried Tel.: 089/8 57 2047
SYSDAT CH-3627 Heimberg Tel.: 033/37 70 40
INDUTRONIC A-9020 Klagenfurt Tel.: 0422/4 36 93-0

News

Endlich ein MS-DOS Emulator für den ATARI ST

Nicht ATARI, sondern ein deutscher Entwickler hat es geschafft, einen MS-DOS Emulator für die ATARI ST Serie herzustellen. Die Firma BETA Systems AG bietet allen ATARI Anwendern ab Oktober ein Gerät an, daß die Möglichkeit bietet, mit dem ST einen IBM XT zu simulieren. Das Konzept stellt eine Mischung aus Hard- und Software dar, die zusammen eine völlige IBM Kompatibilität ausmachen.
Die eigentliche MS-DOS Implementation besteht aus zwei Teilen. Einem 8086-seitigen Teil, der ein IBM kompatibles Entry-Point-Image (EPI) bereitstellt, also vom Verhalten her einem IBM-PC völlig gleicht (inklusive der BIOS Einsprungadressen und aller verfügbaren Register), und einem 68000-seitigem Teil, der die Reaktionen auf Interaktionen mit dem EPI ausführt, sofern sie sich auf Peripherie bezieht, die sich auf Seite des ATARI's befinden. Da einige MS-DOS Programme nicht „sauber" programmiert sind, werden solche vom SUPERCHARGER (so heißt dieser Emulator) abgefangen und der Treibersoftware gemeldet. Diese simuliert dann eine „virtuelle Hardware", oder setzt sie auf die ATARI Hardware um.
Die gesamte Hardware ist auf einer sehr kleinen Platine (etwa nur 10x10 cm groß) untergebracht. Auf dieser befindet sich der Prozessor 8086, 1 MByte RAM und ein hochintegrierter Custom-Chip, den die Firma selbst designed hat. Die 1 MByte RAM Kapazität dieser Karte können auch vom ST z.B. als RAM Disk jederzeit benutzt werden. Der Anschluß zum ST erfolgt über die DMA-Schnittstelle. Der SUPERCHARGER unterstützt sämtliche ATARI-Peripherie, einschließlich hochauflösendem Monitor und Festplatte, so daß auch für den MS-DOS Bereich keine zusätzlichen Geräte notwendig sind.
Der Custom-Chip kennt neben dem Kommunikationsmodus mit der ATARI DMA Schnittstelle noch einen weiteren, flexibleren und schnelleren Modus, der sogenannten TROL Schnittstelle, um mit weiteren SUPERCHARGER Karten ein Netzwerk zu bilden.

Die wichtigsten Daten des SUPERCHARGER

- Eigenständiger 8086 Rechner
- IBM CGA Unterstützung für ATARI Farb- oder Schwarzweiß-Monitor
- IBM kompatibles Diskettenformat
- Erweiterbar (z.B. 8087 oder 80286)
- Maus Unterstützung
- TROL Schnittstelle
- Parallel Verarbeitung
- Multitasking
- Multiuser
- MS-DOS 3.2
- IBM-BIOS Interrupts

Die Hard- und Software des SUPERCHARGERS wird für unter DM 700,— zu haben sein. Sobald dieses interessante Produkt, von dem wir uns schon persönlich überzeugt haben, auf dem deutschen Markt erhältlich ist (vorraussichtlich Ende Oktober), werden wir einen ausführlichen Bericht veröffentlichen.

NEC-EMU EPSON FX/JX-80 EMULATOR jetzt auch für die ST Rechner

Nachdem wir in der Ausgabe Juli/August '87 über einen Epson FX/JX-80 Emulator für die Pinwriter Serie von NEC berichtet hatten, liefen bei NEC in München die Telefone heiß, und der Briefträger brauchte tatkräftige Unterstützung. Leider war dieser Emulator nur für den IBM PC und Kompatible gedacht. Aber nach der großen Nachfrage wurde bei NEC sofort eine Version für den ATARI ST geschrieben, wofür wir uns im Namen unserer Leser ganz herzlich bei dieser Firma bedanken wollen. Der Emulator sollte bei allen NEC-Fachhändlern erhältlich sein und kann natürlich auch bei uns im Public Domain Service (Diskette 88) bezogen werden.

NEC-Deutschland GmbH
Klausenburger Straße 4
8000 München 80
Tel.: 089/93006-0

Nec Emu

BTX am ST

Bildschirmtext am ATARI ST

Die Frankfurter Firma Tommy Software bietet für den ATARI ST mit monochromen Monitor unter dem Macintosh Emulator ALADIN ein Softwarepaket an, das den ATARI ST zu einem BTX-Terminal verwandelt. Benötigt wird dazu lediglich ein Bildschirmtext-Decoder der Deutschen Bundespost und das Programmpaket BTX ST von Tommy Software.

BTX ST ermöglicht das direkte Abspeichern von Bildschirmseiten und Texten. Dem Anwender steht u.a. die Möglichkeit zur Verfügung, Tastaturmakros zu definieren.

BTX ST wird als reine Softwarelösung zum Preis von DM 398,— oder als Komplettpaket (Software und spez. Akustikkoppler mit Kabel) geliefert (Preis auf Anfrage). Ferner bietet Tommy Software einen Beratungsservice für die Beantragung eines Abschlusses an.

TRASH HEAP

Ebenfalls bietet die gleiche Firma ein 3D-Spiel mit dem Namen Trashheap an. „Zieh die 3D-Brille über, mach das Licht aus und bleib cool sitzen".

So fängt die Anleitung dieses exotischen Spieles an, das unmöglich zu beschreiben ist. Das Programm wird am Mitte September lieferbar sein. Zum Lieferumfang gehören eine 3D-Brille und zwei einseitige Disketten.

TrashHeap läuft auf jedem ATARI ST mit mindestens 1MB RAM. Ebenso ist dieses Spiel in Farbe und S/W lauffähig. Der Preis beträgt DM 98,—.

Damit setzt Tommy Software seine erfolgreiche Serie von Spiele-Software fort.

Tommy Software
Gutzkowstr. 35
6000 Frankfurt 70
Tel.: 069/61 40 46

Mirage — Ein Multitasking-Multiuser Betriebssystem für den ST.

Ein neues Multitasking — Multiuser Betriebssystem für den ST

MIRAGE ist nicht nur der Namen eines französischen Kampfflugzeuges oder eines Sample-Keyboards der Firma Ensoniq, sondern auch der Name eines netzfähigen Multitasking — Multiuser-Betriebssystem für den ATARI ST. MIRAGE wurde Anfang der 80'er Jahre von dem englischen Softwarehaus Swifte Computer System entwickelt und ist völlig an das Konzept der MOTOROLA 68xxx-Familie angepaßt. MIRAGE wurde komplett in Assembler geschrieben, um ein Maximum an Geschwindigkeit zu erreichen.
Unter dem MIRAGE-Betriebssystem laufen bereits mehrere Hochsprachen, sowie Anwender-Programme, so daß der Programmierer nicht verlassen auf weiter Flur steht.
Folgend einer Übersicht der MIRAGE ST Leistungsmerkmale:
- Multiuser, Multitasking und Netzwerkfähige
- Real-Time Scheduler. Sehr Kompakt
- Unix-ähnliches Betriebssystem. Extrem schnell und vielseitig
- Liest und schreibt TOS-Dateien.
- Unterstützt Line-A Grafik.
- Mit BASIC und Pascal in Run-Time Systeme
- Deutsche Anpassung

MIRAGE wird als ROM-MODUL geliefert. Ein Ausführliches Handbuch gehört zu Lieferumfang. Der Preis beträgt DM 348,—.

GDAT
Stapelbrede 39
4800 Bielefeld 1
Tel.: 0521/875888

PROLOG für Einsteiger

Ein MProlog Einsteigerpaket wird von der Firma epsilon aus Berlin für den ATARI-Anwender angeboten. Dieses Einsteigerpaket bietet den gesamten Sprachumfang eines professionellen Prolog an. Über 150 Prädikate wurden eingebaut. Ebenso wird eine Programmentwicklungsumgebung mit dedizierem Prolog-Editor sowie Trace und Debugmöglichkeiten geboten.
Die mit diesem Prolog Einsteigerpaket entwickelten Programme sind syntaktisch kompatibel zum de-facto Prolog-Standard, der im Buch „Programming in Prolog" von Cloksin und Mellish beschrieben wurde.
Die eingebaute 3D Eagle-Graphic MPrologs ermöglicht die Gestaltung aussagekräftiger Benutzerschnittstellen für jeden Anwendungszweck.
Sollen die mit dem Einsteigerpaket entwickelten Programme später kommerziell genutzt werden, so können der MProlog Pretranslator und Consolidator nachgerüstet werden. Diesen beide Erweiterung ermöglichen das Erzeugen von stand-alone Programmen, bei denen der Endbenutzer keinen Zugriff mehr auf den Quellcode des Programms hat. Das MProlog Einsteigerpaket kostet DM 398,—, der Pretranslator und Consolidator DM 600,—.

epsilon
Gesellschaft für Software
und Systementwicklung mbH
Kurfürstendamm 188/189
1000 Berlin 15
Tel.: 030/8826991

M Prolog ‚Eagle Graphic' bei der Anwendung im Expert-System.

Neues von Application Systems /// Heidelberg

Unter dem Namen **SiFoX** (Signum Font Exchange) bietet Application Systems eine Reihe von Zeichensatzdisketten für jeweils DM 30,— für alle registrierten Benutzer von Signum gegen Angabe der Registriernummer an. Dabei werden diese Disketten immer für bestimmte Druckertypen zusammengefaßt. SiFoX 1, 3 und 4 enthalten 24 Nadler-Zeichensätze und SiFoX 2 Zeichensätze für 9-Nadeldrucker. Zusätzlich gibt es eine Notenzeichensatz-Diskette.
Man kann eine SiFox-Diskette aber auch gegen Einsendung eines selbstkreierten Zeichensatzes im Austausch erhalten.
Zusätzlich kann man neuerdings die Zeichensatz-Diskette **Julia** mit elf Zeichensätzen für 24- und 9-Nadeln erwerben.
Diverse Laserdruckeranpassungen wie z.B. für HP-Laserjet oder Kyocera F1010 sind ebenfalls erhältlich.

Application Systems /// Heidelberg
Englerstr. 3
6900 Heidelberg
Tel.: 06221/300002

CALAMUS ein Satz und Layoutprogramm für den ST

Calamus ist ein Satz- und Layoutprogramm, das die elektronische Gestaltung von Seiten und Dokumenten ermöglicht. Die Firma DMC (Design Marketing Communication) stellt mit diesem Produkt ein neues Konzept in der Welt des Desktop Publishings für den ATARI ST vor.
Durch den Einsatz von aufwendiger Softwaretechnik wurde ein Produkt geschaffen, das sonst nur auf den teuersten Fotosatzanlagen zu finden ist.
Dazu gehört insbesondere die Verwendung von Vektorzeichensätzen, sowohl bei der Bildschirmausgabe, wie auch im Druck. Sie erlaubt die Darstellung von 16 unterschiedlichen und gleichzeitig verfügbaren Zeichensätzen in nahezu stufenlosen Größen von 4 bis 127,5 Punkte.
Horizontales und vertikales Kerning, Rechtsschreibeüberprüfung und Silbentrennung (automatisch und manuell) sind implementiert. Die piktografische Benutzeroberfläche und die Unterstützung durch Hilftexte und Maus erleichtern den Einstieg in die Welt des Desktop Publishing.
Die Verwendung von PDL Languages (Seitenbeschreibungssprachen) wie Postscript, Interpress, DDL etc. ist möglich und vorgesehen. Das Programm, das völlig in WYSIWYG-Manier funktioniert, besitzt eine eigene Seitenbeschrei-

Calamus — Satz- und Layoutprogramm für den ST.

Aktiver Hub

bungssprache, die genügend Eigenintelligenz hat, um für das jeweilige Ausgabegerät (z.B. den ATARI- oder auch andere Laser- oder Matrixdrucker) die maximale Druckqualität aufzubereiten.
Calamus, das mit deutschem Handbuch geliefert wird, kostet DM 998,—.
Ebenso bietet die Firma DMC einen FX-80i Emulator für den ATARI Laserdrucker.
Laserbrain, so wird dieser Emulator genannt, arbeitet mit allen Programmen, die den Drucker über TOS ansteuern können, ohne daß eine weitere Anpassung notwendig wäre. Ebenfalls emuliert er die meisten Escape-Sequenzen und ist grafikfähig. Hardcopies erfolgen in fünf verschiedenen Größen. Das Programm erlaubt den Betrieb des Laserdruckers mit leichten Einschränkungen auch an einem ST 520.
Das Programm, das mit einer ausführlichen Anleitung geliefert wird, kostet DM 298,—.

DMC Design Marketing Comunication
Schöne Aussicht 41
6229 Walluf
Tel.: 06123/73881

Aktiver Hub für Arcnet Netzwerke

Der Hersteller von Arcnet Netzwerke, Siebert Computersysteme, präsentiert jetzt ein neues Produkt für die ST Rechner. Es handelt sich hierbei um eine preisgünstige Lösung, Arcnet Netzwerkkarten zu verbinden. Der Aktive Hub (SC-HUB) enthält acht BNC-Buchsen, die die Verbindung zu den Netzwerkkarten herstellen. An diesen Buchsen sind jeweils ein Empfänger und ein Sender geschaltet, die durch ein intelligentes Multiplexverfahren miteinander verbunden werden.
Mit dieser einfachen Lösung können bereits installierte Netzwerke miteinander gekoppelt werden.
Der Einzelpreis des Aktiven Hub's beträgt DM 1398,—.

Siebert Computersysteme
Wittelsbacher Straße 80
5040 Brühl
Tel.: 02232/27644

Noch eine Tastatur für den ST

Für die professionelle Arbeit mit dem ATARI ST wurde von der Firma Binnewies eine mechanische Tastatur entwickelt.

Das durchaus ergonomisch geformte Gehäuse wird über ein 2 Meter langes Kabel mit dem Computer verbunden. Neben der gesamten Tastaturlogik sind Maus- und Joystickbuchsen, sowie ein Resettaster auf der Platine enthalten.
Bei den Tasten handelt es sich um mechanische Druckpunkttasten mit 2.5 mm Hub.
Sie wird komplett mit Handbuch und Montagematerial zu einem Preis von DM 389,— geliefert.

Marc Binnewies
Barfeldstraße 37
3000 Hannover 91
Tel.: 0511/431006

LCD Flachdisplay am ATARI ST

Um den ATARI ST transportabel zu machen, bietet jetzt die in Lünen ansässige Firma Wilhelm Mikroelektronik ein LCD Display für den ATARI ST an. Das Display, das in zwei Versionen verfügbar ist, weist folgende gemeinsame Merkmale auf:

- direkt anschließbar an den ATARI ST. Kein Geräteeingriff erforderlich. Wartungsfreies LCD-Display neuester Technologie.
- Hoher Kontrast durch Super Twisted-Technologie mit sehr hoher Lebensdauer.
- Geringe Stromaufnahme und geringes Gewicht.
- Flimmerfrei und damit augenfreundlich wie der Originalmonitor.
- Stellt den Monochrommodus mit voller Auflösung (640x400) dar und ist dadurch kompatibel mit aller Software, die in diesem Modus arbeitet.

Eine Version, die mit einem Schwenkarm versehen ist, ist für den Einsatz im Büro gedacht und wird von der Rückseite beleuchtet, so daß das Bild extrem kontraststark bleibt. Bei dieser Version wird ein Koffer mitgeliefert, der den ATARI ST mit Diskettenlaufwrk und Flachdisplay aufnehmen kann.
Bei der zweiten Version handelt es sich um ein durchleuchtbares Display für einen Tageslichtschreiber. Dadurch wird zum ersten Mal das Bild des Monochrommonitors an die Wand projizierbar. Wegen seiner leichten Handhabung und Portabilität eignet er sich für den Einsatz in Schulen und Universitäten.
Preise waren bis zum Redaktionschluß nicht bekannt.
Ferner bietet die gleiche Firma eine RAM-Disk der neuen Generation an, die durch die Leistungsmerkmale besticht. Die ST-RAM-Disk ist resetfest, in ihrer Größe veränderbar (ähnlich wie die dynamische RAM-Disk beim AMIGA) und sehr schnell. Durch geschickte Programmierung lassen sich mehrere resetfeste RAM-Disks installieren. Eine im Programm integrierte Routine überprüft laufend den Inhalt der RAM-Disk auf Fehler und zeigt diese sofort an. Nach einem Totalabsturz wird somit das Risiko des Datenverlusts vermindert. Programme, die eventuell Schaden genommen haben, werden mit ihrem Namen angezeigt.
Sie läuft auf allen ST Modellen, also auch auf den 4 MegaByte-Rechnern mit Blitter-los und ist ebenso kompatibel mit Harddisktreiberprogrammen. Die Lieferung erfolgt auf Markendisketten und enthält eine Bedienungsanleitung. Der Preis beträgt DM 45,—.

Wilhelm Mirkoelektronik
Büggelstr. 31
4670 Lünen
Tel.: 02306/14025

Keyboard-Editor

Ein Programmpaket, das das Kreieren einer eigenen Zeichenbelegung der Tastatur ermöglicht, wird von Schlegel Datentechnik angeboten. Ebenso können fast alle Tasten mit Texten bis zu einer Länge von 76 Zeichen belegt werden. Diese Texte werden später mit einer wahlfreien Kombination der Tasten Control, Alternate und Shift ausgelöst. Für Programmierer ist solch ein Programm sehr interessant, da z.B. die Schlüsselworte einer Programmiersprache auf diese Weise via Tastendruck verfügbar sind. Der Preis für das Paket beträgt DM 39,—.
Ein weiteres Produkt dieser Firma ist eine sogenannte Datei-Auswahl-Box, die die Fileselectbox des Betriebssystems erweitert. Das Programm wird als Accessory geladen, wobei es sich automatisch im GEM installiert. Dieses Accessory wird im Paket bestehend aus zwei Versionen geliefert. Die eine Version erzeugt einen Eintrag im Desk-Menü und ist ein- und ausschaltbar. Die andere dagegen erzeugt keinen Eintrag und bleibt somit nach dem Aufruf immer resident. Der Preis für dieses Accessory beträgt DM 39,—.

Schlegel Datentechnik
Schwarzachstraße 3
7940 Riedlingen
Tel.: 07371/2317

Neues Netzwerk für den ATARI ST

Die Firma Biodata in Niedernhausen bietet ein Netzwerk für alle ST Rechner an, mit dem bis zu 3200 ATARI's vernetzt werden können. Die Übertragung erfolgt mit kostengünstigen Coaxialkabeln über die DMA-Schnittstelle mit einer Übertragungsgeschwindigkeit von 1 Mega Bit/sec. Im File-Server sind Platten bis zu 300 MByte und schnelle Streamer integrierbar.
Eine Weiterentwicklung (bionet 02) mit eigenem Multitasking/Multiuser-Betriebssystem, eigenem ISAM baut auf dem Ethernet-Standard auf und wird im Oktober dieses Jahres angeboten. Die Übertragung erfolgt hier mit 10 Mega Bit/sec. Es lassen sich Platten bis zu 2 Giga Byte und natürlich auch Streamer anschließen. In diesem Netz können neben ATARI's auch andere Computer betrieben werden, die auf dieselben Datenbestände zurückgreifen. Für Record und File-Locking sind Vorkehrungen getroffen.

Biodata GmbH
Herrenweg 23
6272 Niedernhausen
Tel.: 06128/73001

FIBUMAN wird erweitert

Das Buchhaltungspaket FibuMAN wird jetzt in der Version 1.4 von Prodata angeboten. Die Neuerungen der Version 1.4 umfassen eine erweiterte, komfortablere und für jeden gängigen Drucker ausgelegte Durckereinstellung mit frei definierbarer Formularlänge. Sie ist für Auswertungen und Kontoauszüge getrennt einstellbar.
Als besonderes Bonbon wurde eine Option für schnelles Blättern in allen buchungstechnisch relevanten Listen (Journal, Kostenrahmen, Kontenplan) auch während des Buchungsvorgangs implementiert.
Die Version 1.4 ist als kostenloses Update für Anwender früherer Programmstufen gegen Einsendung der Originaldisketten, Kopie der Rechnung und eines frankierten Rückumschlags umgehend verfügbar.

Prodata
Brolicherstraße 39
5060 Bergisch Gladbach 1
Tel.: 02204/81456

Bildschirmausdruck „LOGOCO Feuerwehr"

Neue Bausteine für FEUDAL System

Eine Serie von Bausteinen für die computergesteuerte Simulation ‚Feuerwehr im Einsatz wird von Keudel av-Technik geboten. Die neue Serie besteht aus drei Dateien:
Gefährliche Stoffe: Diese Datei bietet zu etwa 10000 Stoffnamen, die für den Einsatzfall wichtige schnelle Kurzinformation.
Feuerwehrverwaltung: Dabei handelt es sich um eine relationale Datenbank, die speziell auf die Belange der Feuerwehrverwaltung abgestimmt ist.
LOGOCO Feuerwehr: Mit diesem Programm lassen sich mühelos eigene Lerneinheiten nach dem Multiple-Choice-Verfahren zusammenstellen.
Sowohl die Umsetzung der Datein ‚Gefährliche Stofe', als auch ‚Feuerwehrverwaltung' wurden mit ADIMENS erstellt.

Keudel av-Technik GmbH
Am Stocker 2
6331 Waldsolms
Tel.: 06085/1707

Eine gute Idee ...

ist der Konzepthalter THINGI. Ein, am Monitor befestigter Plastikarm mit einer Halteklammer erlaubt es, Listings. Anleitung oder sonstige Schriftstücke in eine bessere Sichtposition zu bringen. Dadurch, daß das Schriftstück sich genau neben dem Monitorbild befindet, ist es im direkten Blickfeld und es ist nicht mehr nötig, auf einem Blatt neben der Tastatur herzumzusuchen. Besonders das Abschreiben von Texten und Programmlistings geht somit bequemer und auch schnelle. THINGI ist aus weißem, robustem Kunststoff. Die Befestigung erfolgt mit einem selbstklebenden Klettverschluß, der es gestattet, den Halter einfach abzunehmen. THINGI ist für DM 24,90 zuzüglich MwSt. und Versandspesen bei folgender Adresse zu beziehen:

FRENGER Industrievertretung GmbH
Frankonenweg 5
7410 Reutlingen

1st Patch

– zwei Dutzend Nadeln machen Druck

Kennen Sie das? Da hat man sich nun endlich das langersehnte, ultragrelle Programm angelandet, lädt die erstbeste Demo und ist fasziniert ob der fantastischen Möglichkeiten, die sich einem da auf dem Monitor präsentieren. Der erste Frust stellt sich allerdings meist sehr schnell ein, nämlich dann, wenn man den Drucker anwirft: Gemeinerweise ist unter den Myriaden mitgelieferter Druckertreiber wieder mal keiner vorhanden, der auf das eigene Lieblings-Peripheriegerät zugeschneidert ist. In diesen Fällen gibt es zwei Möglichkeiten: Man kauft sich einen Treiber, so es überhaupt einen gibt, oder aber man macht dem Betriebssystem Beine und schreibt „das Ding“ selbst. Kein Problem, wenn man 1st Patch zur Hand hat...

Wer schon einmal mit dem Gedanken eines Druckerkaufes gespielt hat und sich im Dschungel der 8-, 9-, 18-, 19- und 24-Nadler, der Laser-, Typenrad- und Non-Impact-Drucker umgetan hat, wird es messerscharf erkannt haben: Vielfalt ist angesagt. Deshalb kann man es den Softwarehäusern eigentlich nicht verübeln, wenn sie ihre ach so kostbare Entwicklungszeit lieber mit dem Schreiben der eigentlichen Programme denn mit der Erstellung von Hunderten von Druckertreibern verbringen. In aller Regel kann man ja auch mit gemäßigtem Aufwand die mitgelieferten Druckertreiber anpassen, wobei man beispielsweise das Textverarbeitungsprogramm WORDPLUS in punkto Anpassungsfreundlichkeit lobend hervorheben muß. Auch die Möglichkeit, Grafiken in den Text einstreuen zu können, besticht durch überlegene Eleganz. Dann aber schlafen einem doch glatt die Füße bei der Vorstellung ein, daß eben jenes gerade noch so positiv erwähnte Produkt im Zeitalter der LQ-Drucker nur so acht bis neun Nadeln aus ihrem Magnetbett reißt. Da hilft denn auch die beste Druckeranpassung nicht mehr weiter, wenn das Programm selbst die Schönschreiber boykottiert. Man könnte ja nun der Idee verfallen, sich ein eigenes Ausgabeprogramm zu schreiben, oder

```
 1:   ; ##########################################
 2:   ; #          1st Patch für TRAP 1          #              25.06.87
 3:   ; ##########################################
 4:
 5:   ; Dieser Druckertreiber klinkt sich ins GEMDOS ein und verändert
 6:   ; die folgenden Steuercodes (Escape-Sequenzen):
 7:   ;     * ESC 2A 04 (Einschalten der CRT I-Grafik) wird umgewandelt
 8:   ;                 in ESC 2A 39..., wodurch die 24-Nadelgrafik mit
 9:   ;                 dreifacher Dichte eingeschaltet wird. Alle nach
10:   ;                 diesem Kommando  folgenden  Grafikbytes  werden
11:   ;                 bitweise verdreifacht (24 Nadeln!), doppelt ge-
12:   ;                 sendet und jedes 4. Byte sogar dreifach, um die
13:   ;                 gleiche Breite wie bei CRT I zu erhalten.
14:   ;     * FS 33 n   ist für den NEC P6 das Steuerzeichen zum n/360"
15:   ;                 Zeilenvorschub.  1st Patch wandelt 1C (= FS) in
16:   ;                 1B (= ESC) um und halbiert n, wodurch der glei-
17:   ;                 che Zeilenvorschub (=(n/2)/180") erreicht wird.
18:   ;     * FS 46     (Papiervorschub) wird völlig ignoriert, da kei-
19:   ;                 ne vergleichbare Funktion existiert.
20:
21:    text
22:
23:    pea       PATCH               ; Adresse der neuen TRAP #1-Routine
24:    move.w    #33,-(a7)           ; Vektor #33...
25:    move.w    #5,-(a7)            ; ...mittels Setexec...
26:    trap      #13                 ; ...umbiegen
27:    addq.l    #8,a7               ; Stack aufräumen
28:    move.l    d0,AD_GDOS          ; alten Vektor merken
29:    clr.w     -(a7)               ; kein Fehler aufgetreten
30:    move.l    #2000,-(a7)         ; 2000 Bytes reservieren
31:    move.w    #$31,-(a7)          ; KEEP PROCESS
32:    trap      #1                  ; ...und weggetreten!
33:   ;==============================================================
34:   PATCH:
35:    move.l    #-1,DO_ZWS          ; Default auf ok
36:    movea.l   a7,a0               ; SSP -> A0
37:    btst      #5,(a0)             ; Aufruf aus S-Mode
38:    beq.s     FROM_USER           ; nein
39:    addq.l    #6,a0               ; ja: Offset addieren
40:    bra.s     IS_IT_PRT
```

in WORDPLUS selbst ein paar Bytes abzuändern, jedoch sind die Aussichten auf Erfolg speziell in letzerem Fall äußerst gering. Selbst wenn die Operation wider Erwarten gelingen sollte, so funktioniert es doch nur bei diesem einen Programm. Viel aussichtsreicher und universeller ist folgende Überlegung: Wenn ein Programm auf die Centronics-Schnittstelle zugreift, so passiert dies in aller Regel über eine Betriebssystemfunktion des GEMDOS, namentlich der Funktion Printer Output, GEMDOS 0x5. Dann müßte es doch möglich sein, das auszugebende Byte auf seine Wirkung zu untersuchen und gegebenenfalls ein paar ganz andere Zeichen an den Drucker zu senden. Sie werden es vermuten: Es ist wirklich möglich – sofern besagtes Programm auch tatsächlich über das Betriebssystem mit dem Drucker kommuniziert und nicht etwa eigene Routinen benutzt.

ESC-Sequenzen

Im Zusammenhang mit WORDPLUS ist der Befehl: ESC 2A 04 n1 n2 DATA interessant, weil er die 8-Nadel-Emulation (CRT I, 640 Punkte/Zeile) einschaltet. Wie Bild 1 verdeutlicht, werden die Bits der nachfolgenden Grafikdaten nur jeder dritten Nadel zugeordnet, wodurch dann die ungeliebten Lücken entstehen. Abhilfe kann hier durch Einschalten des 24er Modus mit dreifacher Dichte geschaffen werden (ESC 2A 27). Hier müssen dann allerdings pro Spalte gleich drei Bytes vorstellig werden, wozu man die von WORDPLUS abgeschickten Daten verdreifacht. Dies muß aber bitweise passieren, weil man sonst dieselbe Zeile druckt wie im CRT I-Modus (jetzt halt mit 24 Pins), auf ein Drittel der ursprünglichen Höhe gestaucht. Dafür aber dreimal untereinander. Der verfremdende Effekt läßt sich auf Dauer ziemlich schwer verheimlichen...

Falsche Verhältnisse

Nicht nur vertikal nimmt die Informationsdichte bei 24 Nadeln zu, sondern auch in der Breite, denn um eine Zeile komplett einzuschwärzen, braucht man nun genau 1440 Punkte horizontal, also zweieinviertel mal soviel als im CRT I-Modus.

Trägt man dieser Tatsache keine Rechnung, erscheint der mitgelieferte Demo-Tiger zwar in hervorragender Briefqualität; es drängt sich jedoch der Ein-

```
 41:   FROM_USER:
 42:    move.l   USP,a0              ; USP benutzen
 43:   IS_IT_PRT:
 44:    cmpi.w   #5,(a0)             ; Printer Output?
 45:    beq.s    LOS
 46:    move.l   AD_GDOS,-(a7)       ; sonst Originaladresse
 47:    rts                          ; benutzen
 48:
 49:   LOS:
 50:    movem.l  d1-d7/a0-a6,-(a7)   ; Register retten
 51:    tst.b    ESC_ON              ; ESC-Sequenz eingeschaltet?
 52:    bne.s    MORE_ESC            ; ja
 53:    tst.b    GRF_ON              ; Grafik eingeschaltet?
 54:    beq      TEST_IF_ESC         ; nein, testen auf ESC
 55:    move.w   2(a0),d0            ; auszugebendes Zeichen -> d0
 56:    bsr      TRIPLE              ; dessen Bits verdreifachen
 57:    tst.b    GRF_CNT             ; Zähler für Füllbytes schon 0?
 58:    bne.s    \los_end            ; nein, normal weiter
 59:    bsr      OUT_3               ; sonst Byte einmal ausgeben
 60:    move.b   #4,GRF_CNT          ; Zähler wieder rücksetzen
 61:   \los_end:
 62:    bsr      OUT_3               ; Zeichen doppelt ausgeben
 63:    subq.b   #1,GRF_CNT          ; Zähler für Füllbytes dekrementieren
 64:    subq.w   #1,GRF_BYT          ; Anzahl zu sendender Bytes dekrement.
 65:    bne      PATCH_END           ; noch nicht fertig
 66:    clr.b    GRF_ON              ; sonst Grafik-Flag löschen
 67:    bra      PATCH_END           ; und zurück
 68:
 69:   MORE_ESC:
 70:    cmpi.b   #1,BYT_NR           ; 1. Zeichen nach ESC bzw. FS?
 71:    bne.s    ESC_Z2              ; nein, vielleicht 2.?
 72:    move.b   3(a0),ESC_1         ; Zeichen retten
 73:   ESC_Z1:
 74:    tst.b    FS_ON               ; FS eingeschaltet?
 75:    beq      \esc_z1_esc         ; nö
 76:    cmpi.b   #'3',ESC_1          ; FS 3? (=n/360" Zeilenvorschub)
 77:    beq      \esc_z1_end         ; ja, 3.Byte (=n) abwarten
 78:    cmpi.b   #'F',ESC_1          ; FS F? (=Papiervorschub)
 79:    bne      \esc_z1_send_fs     ; nein
 80:    sf       ESC_ON              ; sonst einfach so tun,
 81:    sf       FS_ON               ; als wäre gar nichts
 82:    sf       BYT_NR              ; gesendet worden
 83:    bra      PATCH_END
 84:   \esc_z1_send_fs:
 85:    move.b   #$1C,d0             ; FS-Code senden (ob's gut ist?)
 86:    bsr      OUT
 87:    bra      NOT_ESC             ; und akt. Zeichen ebenso
 88:   \esc_z1_esc:
 89:    cmpi.b   #$2a,ESC_1          ; ESC 2A?
 90:    bne      NOT_ESC             ; nein, abbrechen
 91:   \esc_z1_end:
 92:    addq.b   #1,BYT_NR           ; # Zeichen erhöhen
 93:    bra      PATCH_END           ; und fertig
 94:   ESC_Z2:
 95:    cmpi.b   #2,BYT_NR           ; 2. Zeichen?
 96:    bne      ESC_Z3              ; nein, vielleicht 3.?
 97:    move.b   3(a0),ESC_2         ; Zeichen retten
 98:    tst.b    FS_ON               ; FS eingeschaltet?
 99:    beq      \esc_z2_esc         ; nein
100:    move.b   3(a0),d0            ; n -> d0
101:    asr.b    #1,d0               ; div 2 (wg. n/360=(n/2)/180)
102:    move.b   d0,ESC_2            ; und abspeichern
103:    move.b   #$1B,d0             ; ESC statt FS
104:    bsr      OUT                 ; ausgeben
105:    bra      NOT_ESC             ; und den Rest auch
106:   \esc_z2_esc:
107:    cmpi.b   #4,ESC_2            ; ESC 2A 04 (CRT I ein)?
108:    bne      NOT_ESC             ; nein, abbrechen
109:    move.b   #39,ESC_2           ; sonst 24er-Grafik-Code speichern
110:    addq.b   #1,BYT_NR           ; # Zeichen erhöhen
111:    bra      PATCH_END           ; und fertig
112:   ESC_Z3:
113:    cmpi.b   #3,BYT_NR           ; 3. Zeichen?
114:    bne.s    ESC_Z4              ; nein, dann 4. Zeichen
115:    move.b   3(a0),GRF_LO        ; n1 speichern
116:    addq.b   #1,BYT_NR           ; Index erhöhen
117:    bra      PATCH_END           ; und fertig
```

Bild 1: Links: So druckt WORDPLUS Grafiken...
Rechts: Und das macht 1st Patch daraus

```
118:    ESC_Z4:
119:      st        GRF_ON              ; Grafik einschalten
120:      sf        ESC_ON              ; ESC-Modus aus
121:      move.b    #3,GRF_CNT          ; Zähler für Füllbytes initialisieren
122:      move.b    3(a0),GRF_HI        ; n2 speichern
123:      move.b    ESC_1,d0            ; 1. Zeichen ausgeben
124:      bsr       OUT
125:      move.b    ESC_2,d0            ; 2. Zeichen ebenso
126:      bsr       OUT
127:      move.w    GRF_BYT,d0          ; Anzahl Grafik-Bytes speichern
128:      move.w    d0,d1               ; in d1 retten (-> Füllbytes berechnen)
129:      add.w     d0,d0               ; n*2 (wegen Doppeldruck)
130:      asr.w     #2,d1               ; n durch 4 dividieren
131:      add.w     d1,d0               ; und zu 2n addieren
132:      bsr       OUT                 ; n1 ausgeben
133:      ror.w     #8,d0               ; Bytes tauschen
134:      bsr       OUT                 ; n2 ausgeben
135:      tst.w     d0                  ; keine Grafikdaten?
136:      bne.s     ESC_END             ; dann Schluss
137:      clr.b     GRF_ON              ; und Grafik löschen
138:    ESC_END:
139:      bra       PATCH_END           ; und fertig
140:
141:    NOT_ESC:
142:      sf        ESC_ON              ; war nix von wegen CRT I ein
143:      sf        FS_ON               ; und auch nix mit n/360"
144:      move.b    ESC_1,d0            ; 1. Byte nach ESC
145:      cmpi.b    #1,BYT_NR           ; erst beim 2. Byte als ungültig erkannt?
146:      beq       \not_esc_end        ; nein: fertig
147:      bsr       OUT                 ; sonst 1. und
148:      move.b    ESC_2,d0            ; 2. Byte nach ESC ausgeben
149:    \not_esc_end:
150:      sf        BYT_NR              ; Anzahl Zeichen nach ESC/FS rücksetzen
151:      bra       CHAR_OUT            ; Zeichen in d0 ausgeben und beenden
152:
153:    TEST_IF_ESC:
154:      move.w    2(a0),d0            ; auszugebendes Zeichen -> d0
```

druck auf, als hätte er seinen Kopf versehentlich auf einer Dampframme abgelegt... 1st Patch behebt diesen Mißstand dadurch, daß jede Pixelspalte doppelt und jede vierte dreifach ausgegeben wird. Ergebnis: Der Tiger packt's wieder.

NEC-ereien

Zum neuen Star am LQ-Grafik-Himmel hat sich aufgrund seiner überragenden Leistungen die NEC Px-Reihe gemausert. Für sie sind Pixelabstände von 1/360 Zoll horizontal und vertikal eine Leichtigkeit. Horizontal sind z.B. Epson-Drucker durchaus kompatibel; beim Zeilenvorschub sieht es da etwas anders aus: Bei NEC hat man, um die Fähigkeiten der eigenen Drukker auszukitzeln, das für alle anderen unbekannte Steuersymbol FS (0x1C) eingeführt. Gefolgt von: 33 n befiehlt es dem Drucker, das Papier um n/360" voranzutreiben. Da wohl jeder Drucker die ESC-Sequenz 1B 33 n versteht (Zeilenvorschub um n/180"), fängt 1st Patch das NEC-Kommando ab und bedient sich des uralten mathematischen Prinzips der Kürzung: Wenn x=n/360, dann auch x=(n/2)/180. Außerdem wird die Sequenz FS 46 von 1st Patch abgefangen und einfach ignoriert; sie soll wohl auch eine Art Zeilenvorschub erzeugen, ihr Fehlen stört aber nicht weiter. Alle anderen FS- und ESC-Sequenzen werden von 1st Patch unverändert weitergegeben; notfalls muß man sie halt ebenfalls abfangen und ändern (s. ESC__Z1, ESC__Z2). Wer einen NEC Px besitzt, sollte natürlich die mit (★) markierten Zeilen im Programm weglassen bzw. in der angegebenen Form modifizieren!!!

Wo bitte geht's zum GEMDOS?

Nachdem nun die Problematik des Patches klar zutage getreten ist, stellt sich die Frage, wie man denn ein im ROM befindliches TOS abändern kann. Dies birgt de facto keine Schwierigkeiten: Die Adresse nämlich, wo denn das Betriebssystem seine Traps versteckt hat, liegen im RAM und können bequemerweise sogar mit einer BIOS-Funktion auf die eigenen Routinen umgebogen werden. In D0 erhält man die ursprüngliche Adresse (die man sich merken muß!) zurück. Ab jetzt landen also alle GEMDOS-Aufrufe in unserer Routine. Hier müssen wir nun prüfen, ob der Aufruf aus dem User-

Oh!

Die Kunst der (ST-)Software:

Auweia. Erklärungen abgeben. Software – ich lehne mich zurück. Was macht die neue Software? Danke, sie entwickelt sich – UndSTADundSignum!Was Neues? – So hören wir jeden Tag Anfragen der ST-Computeranwender, die uns schreiben oder uns anrufen. Und es ist ja auch an der Zeit, wieder einmal etwas zu vermelden. Andererseits: ein denkbar ungünstiger Zeitpunkt. Manches ist noch nicht definiert oder definitiv, aber Anzeigen haben auch Vorlaufzeiten. Am liebsten sprächen wir durch eine Ladung Trockeneis. Nun:

Gelbgesichtig noch immer **STAD,** 179,– DM das Grafikprogramm, das fast keine Wünsche offen läßt. Der Kreis seiner geneigten Benutzer vergrößert sich ständig. Ebenso der unserer Textverarbeitung **Signum!**, 448,– DM, mittlerweile wohl die umfassende Textverarbeitung für den ST! Hier gibt es Neuigkeiten für einen gewachsenen Kreis unterschiedlichster Anwender: 3 Sifox*-Disketten, eine Eurofontdiskette sowie unsere Fontdiskette „Julia" sind nun erhältlich und erweitern die Möglichkeiten von **Signum!** erheblich. Sprachwissenschaftler verwenden altgriechische Zeichensätze, schreiben hebräisch oder kyrillisch damit. Musiker verwenden den Notenzeichensatz, Physiker den mathematisch-physikalischen. Unproportionale Schrift ist hilfreich beim Tabellensatz undsoweiter. Auf Anfrage senden wir gerne Inhaltsverzeichnisse der erhältlichen Fontdisketten.

Im Herbst werden wir zwei weitere Programme vorstellen, die mit Grafik zu tun haben oder mit der Darstellung von Grafik: Zunächst **Imagic,** ein Programm das in zwei Auflösungen, in Farbe und schwarz-weiß laufen wird. Seine vielfältigen Überblend-Effekte machen es zu einem interessanten grafischen Entwicklungssystem. An Stichworten hören wir: Ultraschnelle Animation (bis 70 Bilder/s), Bildfolgenpräsentation, Entwicklungspaket in sieben Programmen, Steuerung für Videoproduktionen, Trickfilm, Werbung, Präsentation von Business-Grafik und mehr wird noch nicht verraten.

Ebenfalls neu ist und unerläßlich wird **Creator** sein. Das Programm besitzt einen ausgedehnten Zeichenteil, dessen Details wir noch dezent im Nebel verhüllen wollen, sowie ein Animationsprogramm für die professionelle Erstellung von Zeichentrickfilmen.

Über unsere Sprachen informieren nebenstehende Kästen. Im nächsten Monat hören wir noch mehr Neuigkeiten über bedeutende Entwicklungen. Sollten Fragen sein zu den erhältlichen Programmen, zu Rechnerkonfigurationen undsoweiter, so rufen Sie uns an, wir beantworten sie gerne. Rrrrrrrrrrrrring. – **Oh!**

Megamax Modula-2

Als komplettes Entwicklungssystem für Modula-2 besitzt Megamax Modula-2 einen schnellen Ein-Pass-Compiler, Editor und Debugger unter einer komfortablen GEM-Shell. Der Modula-2 Compiler mit integriertem symbolischen Assembler übersetzt über 5000 Zeilen pro Minute in 68000-er Maschinencode.
Resource Construction Program. Zugriff auf alle TOS-, VDI- und AES-Funktionen. „Load Time Linking", d.h. kein separater Linker-Lauf, die Platzvergeudung durch mehrfach vorhandenen Modul-Code entfällt.

*) Signum Font Exchange

Version 2.0

Megamax C

C-Entwicklungssystem, Version 2.0. Single Pass Compiler + Inline Assembler + volle GEM®-Library + GEM® Editor + Linker + Librarian + Unix-Routinen + Make-File Utility + 370 Seiten **Dokumentation in Deutsch** + GEM® Shell + Resource Construction Program mit ICON Editor + bis zu 10-fach schnellere Mathe-Bibliothek + K-Resource von Kuma + GEM®-Window-Library + Anpassung für FPU 68 881 (Mega-ST, Lischka).
595,– DM

Editor Toolbox

beinhaltet einen Editor mit folgenden Features: vollständig GEM®-integriert + bis zu 10 Texte gleichzeitig edierbar + komplette DFU-Funktionen implementiert + verschiedene Schriftgrößen für optimalen Überblick + zahlreiche Block-Kommandos + Funktionstasten frei belegbar + Sprungmarken + Tastaturkommandos + sehr schnelle Bildschirmausgabe + extrem schnelles Suchen/Ersetzen + **vollständig dokumentierte C-Source für Megamax C.**
149,– DM

B-Tree Toolbox

Die B-Tree Toolbox ist eine ISAM-Dateiverwaltung nach dem B-Tree-Verfahren. Sie garantiert kürzeste Zugriffzeiten auf alle Datensätze einer Datei. Selbst größere Datenbanken lassen sich so in Sekundenbruchteilen nach Begriffen durchsuchen. Mit dem Megamax C-Compiler läßt sich die B-Tree Toolbox leicht in C-Programme einbinden.
198,– DM

APPLICATION SYSTEMS HEIDELBERG Englerstraße 3 Postfach 10 26 46 Telefon (0 62 21) 30 00 02 D-6900 Heidelberg. **Vertrieb Europa, Benelux:** PDS, Promenadeplein 107, NL-2711 AB Zoetermeer. **Österreich:** Ueberreuter Handels- und Verlags GmbH, Alserstraße 24, A-1091 Wien. **Schweiz:** Senn Computer AG, Langstraße 31, CH-8021 Zürich. **Nordeuropa:** New World Computer / Atari Danmark, Skanderborgwej 14, DK-8260 Viby J.

APPLICATION SYSTEMS HEIDELBERG

oder Supervisormodus erfolgte, weil die Daten ja auf dem jeweiligen Stack zu finden sind. Erfolgte der Aufruf aus dem S-Mode heraus, muß man darauf achten, erst einen Offset von sechs Bytes auf den Stack zu addieren, weil die CPU beim Antreffen eines Trap-Befehles automatisch die Rücksprungadresse (.L) und das Statusregister (.W) auf den Stack legt. Als nächstes überprüfen wir, ob es sich um die Printer-Funktion 0x5 handelt. Wenn nicht, springen wir an die (hoffentlich gerettete) Originaladresse des GEMDOS. Ansonsten kann unser Patch in Aktion treten. Beachten muß man nur, daß die üblichen Register (D1-D6/A2-A6) gerettet werden. Abschließend werden die Registerinhalte restauriert und unsere Routine mit RTE abgeschlossen. Als Lohn für unsere Arbeit erhalten wir von nun an nur noch wunderschöne 24-Nadel-Grafiken; die Nicht-Besitzer eines NEC-Druckers zusätzlich noch die faire Chance, in Zukunft wenigstens deren Treiber benutzen zu können. Da es weder zeitliche noch speicherplatzmäßige Restriktionen für die Patch-Routine gibt, darf man nach Belieben zusätzliche Steuercodes abfangen und verändern.

Selbstverständlich kann man nach dieser Methode auch zusätzliche Funktionen ins Betriebssystem einschleusen, sich z.B. eine neue File-Selector-Box bauen, oder die Umlaute mittels 1st Patch in ESC-Sequenzen umwandeln (Einschalten des deutschen Zeichensatzes), auf daß das gewohnte Frusterlebnis bei der Desktop-Option „Drucken" zukünftig ausbleiben möge.

M. Schumacher

```
155:     cmpi.b   #$1B,d0          ; = ESC?
156:     bne      TEST_IF_FS       ; nein
157:     st       ESC_ON           ; sonst ESC einschalten
158:     bsr      OUT              ; und ausgeben
159:     move.b   #1,BYT_NR        ; Zähler für # Zeichen nach ESC erhöhen
160:     bra      PATCH_END        ; und beenden
161:
162:    TEST_IF_FS:
163:     cmpi.b   #$1C,d0          ; = FS?
164:     bne      CHAR_OUT         ; nein
165:     st       FS_ON            ; sonst FS
166:     st       ESC_ON           ; und ESC einschalten
167:     move.b   #1,BYT_NR        ; Zähler für # Zeichen nach ESC erhöhen
168:     bra      PATCH_END        ; und beenden
169:
170:    CHAR_OUT:
171:     bsr      OUT              ; Zeichen ausgeben
172:     bra      PATCH_END        ; Routine beenden
173:
174:    TRIPLE:                    ; Bits von d0.b verdreifachen
175:     clr.l    d1               ; hier kommt das Ergebnis hin
176:     moveq.l  #7,d2            ; unser Bit-Nr.-Zähler für d0
177:     moveq.l  #23,d3           ; unser Bit-Nr.-Zähler für d1
178:    \triple_lp:
179:     btst     d2,d0            ; Bit gesetzt?
180:     bne.s    \triple_mal3     ; dann verdreifachen
181:     subq.l   #3,d3            ; sonst übergehen
182:     bra.s    \triple_end
183:    \triple_mal3:
184:     bset     d3,d1
185:     subq.l   #1,d3
186:     bset     d3,d1
187:     subq.l   #1,d3
188:     bset     d3,d1
189:     subq.l   #1,d3
190:    \triple_end:
191:     dbra     d2,\triple_lp
192:     move.l   d1,d0
193:     rts
194:
195:    OUT_3:                     ; 3 Bytes in d0 ausgeben (d0 = x1 23)
196:     movem.l  d0/d4,-(a7)      ; d0/d4 retten
197:     moveq.l  #1,d4            ; Default-Offset = 2 (doppelt ausgeben)
198:     tst.b    GRF_CNT          ; nur einmal ausgeben?
199:     bne.s    \out_3_lp        ; nein
200:     clr.l    d4               ; sonst Offset = 0
201:    \out_3_lp:
202:     movem.l  d0/d4,-(a7)      ; d0 und d4 retten (lokal)
203:     swap     d0
204:     bsr.s    OUT              ; 1. Byte
205:     swap     d0
```

Links Oben ---------------------- Rechts Oben

Links unten ---------------------- Rechts unten

Bild 2: TIGER.IMG ohne 1st Patch

Links Oben ---------------------- Rechts Oben

Links unten ---------------------- Rechts unten

Bild 3: TIGER.IMG mit 1st Patch

```
206:     ror.w     #8,d0
207:     bsr.s     OUT                  ; 2. Byte
208:     ror.w     #8,d0
209:     bsr.s     OUT                  ; 3. Byte
210:     movem.l   (a7)+,d0/d4          ; Register zurück
211:     dbra      d4,\out_3_lp         ; 1 oder 2mal ausgeben
212:     movem.l   (a7)+,d0/d4          ; Register zurück
213:     rts
214:
215:   OUT:                             ; Ausgabe von d0.b
216:     movem.l   d0-d1/a0,-(a7)       ; benötigte Register sichern
217:     move.b    d0,d1                ; Byte nach d1
218:     ext.w     d1                   ; auf Wortlänge bringen
219:     move.w    d1,-(a7)             ; Byte ausgeben
220:     clr.w     -(a7)                ; auf Drucker
221:     move.w    #3,-(a7)             ; Bconout
222:     trap      #13                  ; BIOS
223:     addq.l    #6,a7                ; Stack aufräumen
224:     move.l    d0,D0_ZWS            ; Return-Wert sichern
225:     movem.l   (a7)+,d0-d1/a0       ; Register zurück
226:     rts
227:
228:   PATCH_END:
229:     movem.l   (a7)+,d1-d7/a0-a6    ; Register zurück
230:     move.l    D0_ZWS,d0            ; Return-Wert
231:     rte                            ; ...und fertig
232:
233:     data
234:
235:   ESC_ON:     dc.b 0               ; Flag, ob ESC-Sequenz gesendet wurde
236:   FS_ON:      dc.b 0               ; Flag, ob FS-Sequenz gesendet wurde
237:   GRF_ON:     dc.b 0               ; Flag, ob Grafik eingeschaltet ist
238:
239:     align
240:
241:   GRF_BYT:                         ; dient nur als Offset
242:   GRF_HI:     dc.b 0               ; Anzahl Grafikbytes, MSB
243:   GRF_LO:     dc.b 0               ; Anzahl Grafikbytes, LSB
244:   BYT_NR:     dc.b 0               ; Zähler für ESC-Codes
245:   ESC_1:      dc.b 0               ; 1. Zeichen hinter ESC
246:   ESC_2:      dc.b 0               ; 2. Zeichen hinter ESC
247:   GRF_CNT:    dc.b 0               ; Zähler für Füllbytes
248:
249:     align
250:
251:   D0_ZWS:     dc.l 0               ; Zwischenspeicher für d0
252:   AD_GDOS:    dc.l 0               ; Originaladresse GEMDOS
253:
254:     end
255:
256:
257:
```

Listing 1: Das steckt hinter 1st Patch

Bild 4: Vorher

Bild 5: Nachher

Listing

Übrigens: Für die seltenen Fälle, in denen über das BIOS (Trap #13, Bconout) ausgedruckt wird, muß man 1st Patch folgendermaßen modifizieren:

1. Das BIOS hat die Vektornummer 45 (Änderung in der zweiten Programmzeile)

2. Der Befehl Bconout hat eine andere Struktur als Printer Output. Deshalb muß man IS_IT_PRT ändern:

```
 cmpi.w   #3,(a0)          ; Bconout?
 bne.s    NIX_DRUCK        ; nö, fertig
 tst.w    2(a0)            ; geht Ausgabe auf PRT: (dev=0)?
 bne.s    NIX_DRUCK        ; sieht nicht so aus
 addq.l   #2,a0            ; sonst Offset für Stack addieren
 bra.s    LOS              ; und anfangen
NIX_DRUCK:
 move.l   AD_GDOS,-(a7)    ; Originalroutine des BIOS
 rts                       ; benutzen
```

3. Da es nicht möglich ist, innerhalb einer Trap-Routine nochmal dieselbe Trap zu aktivieren, muß das Unterprogramm 'OUT' ab der Stelle: trap #13 (incl. derselben) folgendermaßen geändert werden:

```
 pea      \out_ret         ; Rücksprungadresse auf Stack
 move.w   SR,-(a7)         ; Status auf Stack (TRAP-Simulation)
 move.l   AD_GDOS,-(a7)    ; Adresse Originalroutine
 rts                       ; anspringen
\out_ret:
 addq.l   #6,a7            ; Stack wieder aufräumen
 move.l   d0,D0_ZWS        ; Ergebnis der Ausgabe merken
 movem.l  (a7)+,d0-d1/a0   ; Register restaurieren
 rts                       ; und zurück
```

Damit ist auch die Druckausgabe über BIOS nicht mehr vor 1st Patch sicher. 1st Patch wird sinnvollerweise in den AUTO-Ordner der Boot-Diskette gelegt und bleibt dann für die Einschaltdauer resident.

```
  1:   Open "O",#1,"1STPATCH.PRG"
  2:   Cls
  3:   Fertig$="Ende"
  4:   Do
  5:     Read Dat$
  6:     Inc Cnt%
  7:     Exit If Instr(Dat$,Fertig$,1)>0
  8:     Dat$=Right$(Dat$,Len(Dat$)-1)
  9:     If (Cnt% Mod 12)=1
 10:       Lin%=Val(Dat$)
 11:       Chk%=0
 12:       Print At(38,25);Lin%;
 13:     Else
 14:       Dat$="&"+Dat$
 15:       If (Cnt% Mod 12)=0
 16:         If Chk%<>Val(Dat$)
 17:           Print " => Fehler!"
 18:           Chk%=0
 19:         Endif
 20:       Else
 21:         Add Chk%,Val(Dat$)
 22:         Out #1,Val(Dat$)
 23:       Endif
 24:     Endif
 25:   Loop
 26:   Read Dat$
 27:   If Chk%<>Val("&"+Right$(Dat$,Len(Dat$)-1))
 28:     Print " => Fehler!"
 29:   Endif
 30:   '
 31:   ' ################################################
 32:   Dat_lines:
 33:   Data 00001, 60, 1A, 00, 00, 02, C8, 00, 00, 00, 12, 156
 34:   Data 00002, 00, 00, 00, 00, 00, 00, 00, 00, 00, 00, 000
 35:   Data 00003, 00, 00, 00, 00, 00, 00, 00, 00, 48, 79, 0C1
 36:   Data 00004, 00, 00, 00, 26, 3F, 3C, 00, 21, 3F, 3C, 13D
 37:   Data 00005, 00, 05, 4E, 4D, 50, 8F, 23, C0, 00, 00, 262
 38:   Data 00006, 02, D6, 42, 67, 2F, 3C, 00, 00, 07, D0, 2C3
 39:   Data 00007, 3F, 3C, 00, 31, 4E, 41, 23, FC, FF, FF, 458
 40:   Data 00008, FF, FF, 00, 00, 02, D2, 20, 4F, 08, 10, 359
 41:   Data 00009, 00, 05, 67, 04, 5C, 88, 60, 02, 4E, 68, 26C
 42:   Data 00010, 0C, 50, 00, 05, 67, 08, 2F, 39, 00, 00, 138
 43:   Data 00011, 02, D6, 4E, 75, 48, E7, 7F, FE, 4A, 39, 4CA
 44:   Data 00012, 00, 00, 02, C8, 66, 44, 4A, 39, 00, 00, 1F7
 45:   Data 00013, 02, CA, 67, 00, 01, A0, 30, 28, 00, 02, 22E
 46:   Data 00014, 61, 00, 01, E2, 4A, 39, 00, 00, 02, D1, 29A
 47:   Data 00015, 66, 0C, 61, 00, 01, F8, 13, FC, 00, 04, 2DF
 48:   Data 00016, 00, 00, 02, D1, 61, 00, 01, EC, 53, 39, 2AD
 49:   Data 00017, 00, 00, 02, D1, 53, 79, 00, 00, 02, CC, 26D
 50:   Data 00018, 66, 00, 02, 2C, 42, 39, 00, 00, 02, CA, 1DB
 51:   Data 00019, 60, 00, 02, 22, 0C, 39, 00, 01, 00, 00, 0CA
 52:   Data 00020, 02, CE, 66, 62, 13, E8, 00, 03, 00, 00, 296
 53:   Data 00021, 02, CF, 4A, 39, 00, 00, 02, C9, 67, 00, 286
 54:   Data 00022, 00, 3C, 0C, 39, 00, 33, 00, 00, 02, CF, 185
 55:   Data 00023, 67, 00, 00, 3C, 0C, 39, 00, 46, 00, 00, 12E
 56:   Data 00024, 02, CF, 66, 00, 00, 18, 51, F9, 00, 00, 299
 57:   Data 00025, 02, C8, 51, F9, 00, 00, 02, C9, 51, F9, 429
 58:   Data 00026, 00, 00, 02, CE, 60, 00, 01, D8, 10, 3C, 255
 59:   Data 00027, 00, 1C, 61, 00, 01, B0, 60, 00, 00, DE, 26C
 60:   Data 00028, 0C, 39, 00, 2A, 00, 00, 02, CF, 66, 00, 1A6
 61:   Data 00029, 00, D2, 52, 39, 00, 00, 02, CE, 60, 00, 28D
 62:   Data 00030, 01, B6, 0C, 39, 00, 02, 00, 00, 02, CE, 1CE
 63:   Data 00031, 66, 00, 00, 4A, 13, E8, 00, 03, 00, 00, 1AE
 64:   Data 00032, 02, D0, 4A, 39, 00, 00, 02, C9, 67, 00, 287
 65:   Data 00033, 00, 1A, 10, 28, 00, 03, E2, 00, 13, C0, 20A
 66:   Data 00034, 00, 00, 02, D0, 10, 3C, 00, 1B, 61, 00, 19A
 67:   Data 00035, 01, 64, 60, 00, 00, 92, 0C, 39, 00, 04, 1A0
 68:   Data 00036, 00, 00, 02, D0, 66, 00, 00, 86, 13, FC, 2CD
 69:   Data 00037, 00, 27, 00, 00, 02, D0, 52, 39, 00, 00, 184
 70:   Data 00038, 02, CE, 60, 00, 01, 62, 0C, 39, 00, 03, 1DB
 71:   Data 00039, 00, 00, 02, CE, 66, 12, 13, E8, 00, 03, 246
 72:   Data 00040, 00, 00, 02, CD, 52, 39, 00, 00, 02, CE, 22A
 73:   Data 00041, 60, 00, 01, 46, 50, F9, 00, 00, 02, CA, 2BC
 74:   Data 00042, 51, F9, 00, 00, 02, C8, 13, FC, 00, 03, 326
 75:   Data 00043, 00, 00, 02, D1, 13, E8, 00, 03, 00, 00, 1D1
 76:   Data 00044, 02, CC, 10, 39, 00, 00, 02, CF, 61, 00, 249
 77:   Data 00045, 01, 00, 10, 39, 00, 00, 02, D0, 61, 00, 17D
 78:   Data 00046, 00, F6, 30, 39, 00, 00, 02, CC, 32, 00, 25F
 79:   Data 00047, D0, 40, E4, 41, D0, 41, 61, 00, 00, E4, 48B
 80:   Data 00048, E0, 58, 61, 00, 00, DE, 4A, 40, 66, 06, 36D
 81:   Data 00049, 42, 39, 00, 00, 02, CA, 60, 00, 00, F0, 297
 82:   Data 00050, 51, F9, 00, 00, 02, C8, 51, F9, 00, 00, 35E
 83:   Data 00051, 02, C9, 10, 39, 00, 00, 02, CF, 0C, 39, 22A
 84:   Data 00052, 00, 01, 00, 00, 02, CE, 67, 00, 00, 0C, 144
 85:   Data 00053, 61, 00, 00, AE, 10, 39, 00, 00, 02, D0, 22A
 86:   Data 00054, 51, F9, 00, 00, 02, CE, 60, 00, 00, 44, 2BE
 87:   Data 00055, 30, 28, 00, 02, 0C, 00, 00, 1B, 66, 00, 0E7
 88:   Data 00056, 00, 18, 50, F9, 00, 00, 02, C8, 61, 00, 28C
 89:   Data 00057, 00, 88, 13, FC, 00, 01, 00, 00, 02, CE, 268
 90:   Data 00058, 60, 00, 00, 9C, 0C, 00, 00, 1C, 66, 00, 18A
 91:   Data 00059, 00, 1A, 50, F9, 00, 00, 02, C9, 50, F9, 377
 92:   Data 00060, 00, 00, 02, C8, 13, FC, 00, 01, 00, 00, 1DA
 93:   Data 00061, 02, CE, 60, 00, 00, 7C, 61, 00, 00, 58, 265
 94:   Data 00062, 60, 00, 00, 74, 42, 81, 74, 07, 76, 17, 29F
 95:   Data 00063, 05, 00, 66, 04, 57, 83, 60, 0C, 07, C1, 27D
 96:   Data 00064, 53, 83, 07, C1, 53, 83, 07, C1, 53, 83, 412
 97:   Data 00065, 51, CA, FF, EA, 20, 01, 4E, 75, 48, E7, 517
 98:   Data 00066, 88, 00, 78, 01, 4A, 39, 00, 00, 02, D1, 257
 99:   Data 00067, 66, 02, 42, 84, 48, E7, 88, 00, 48, 40, 36D
100:   Data 00068, 61, 18, 48, 40, E0, 58, 61, 12, E0, 58, 3E4
101:   Data 00069, 61, 0E, 4C, DF, 00, 11, 51, CC, FF, E8, 4AF
102:   Data 00070, 4C, DF, 00, 11, 4E, 75, 48, E7, C0, 80, 46E
103:   Data 00071, 12, 00, 48, 81, 3F, 01, 42, 67, 3F, 3C, 23F
104:   Data 00072, 00, 03, 4E, 4D, 5C, 8F, 23, C0, 00, 00, 26C
105:   Data 00073, 02, D2, 4C, DF, 01, 03, 4E, 75, 4C, DF, 3F1
106:   Data 00074, 7F, FE, 20, 39, 00, 00, 02, D2, 4E, 73, 36B
107:   Data 00075, 00, 00, 00, 00, 00, 00, 00, 00, 00, 00, 000
108:   Data 00076, 00, 00, 00, 00, 00, 00, 00, 00, 00, 00, 000
109:   Data 00077, 00, 02, 12, 18, 1A, 0C, 08, 12, 0E, 0A, 084
110:   Data 00078, 06, 0A, 0C, 0A, 06, 0C, 0C, 0A, 06, 06, 05A
111:   Data 00079, 18, 0A, 0C, 0C, 06, 10, 14, 0C, 06, 0C, 082
112:   Data 00080, 0A, 06, 0A, 06, 08, 08, 06, 0A, 0A, 1C, 066
113:   Data 00081, 0A, 06, 06, 08, 0E, 06, 16, 0C, 12, 06, 06C
114:   Data 00082, 08, 3A, 3E, 10, 00, 00, Ende, 090
```

Listing 2: Wenn's nicht anders geht: Der Basiclader

Elemente der künstlichen Intelligenz

Eine Einführung in Programmiermethoden und Sprachen der KI

6. Teil: Die Blockwelt

Einleitung

Nach dem Kapitel Grammatiken wird es jetzt wohl Zeit, mal wieder etwas mehr in Richtung LISP zu blicken. Schließlich ist das erste Projekt zur Spracherkennung von Terry Winograd (siehe Literaturverzeichnis am Ende dieses Artikels) in LISP geschrieben. Das hier vorgestellte Programm ist ein kleiner Ausschnitt, der einerseits einen Eindruck von der Blockwelt vermitteln, andererseits aber auch einige wichtige Konzepte der Sprache LISP nahebringen soll. Schwerpunktthemen sind die Property-Liste, das LET-binding und die Schablonenbenutzung.

Die Problemstellung

Vielleicht erinnern Sie sich nicht mehr, aber wahrscheinlich haben auch Sie früher einmal mit Bauklötzen gespielt? Da liegen also die Bauklötze vor Ihnen in irgendeiner Anordnung, z. B. der von **Abb 1**. Und nun sagt Ihnen jemand: „Pack den roten Würfel auf den blauen." Was sich simpel anhört und simpel ist, erfordert dennoch eine stattliche Menge von Information und Organisation. Zunächst müssen die Objekte lokalisiert werden. Anschließend erfolgt die Bewegung in der richtigen Richtung, denn der Befehl „Pack den roten Würfel auf den blauen" ist nicht zu verwechseln mit „pack den blauen Würfel auf den roten". Dabei müssen Sie natürlich wissen, daß man auf einer Kugel und einer Pyramide nichts ablegen kann, diese selbst aber sehr wohl auf einen Würfel oder in eine Schachtel transportieren kann. Um die Sache interessant zu machen, sollten Sie nun noch Fragen nach dem **Wie, Weshalb** und **Warum** beantworten. Diese Aufgabenstellung erfüllt das in Listing 1 abgedruckte-XLISP Programm, das in Anlehnung an das Programm von Winston und Horn (siehe Literaturverzeichnis) entwickelt wurde. Das letztere benutzt aber die PROCCLAIM-Funktion, die von XLISP leider nicht zur Verfügung gestellt wird. Sie ermöglicht eine dynamische Symbolbindung, entgegen der sonst in LISP üblichen lexikalischen Bindung. Im originalen SHRDLU-Projekt konnten die Befehle in natürlicher Sprache über die Tastatur eingegeben werden. Das würde den Rahmen dieses Aufsatzes natürlich sprengen. Es wird aber versucht, umgangssprachlich klingende Befehle zu verwenden.

Die Propertyliste

Die zentrale Datenstruktur des Blockweltprogramms ist die Propertyliste. Wie der Name schon sagt, handelt es sich um eine **Liste**, die bestimmte **Eigenschaften** (engl.: Property) eines Objektes enthält. Schauen wir uns den Aufbau einer solchen Propertyliste am Beispiel der Objekte der Blockwelt einmal näher an; z. B. ist da der rote Würfel. Im ersten Teil des Programms wird dieses Objekt mit Hilfe folgender Anweisungen definiert:

```
(putprop 'a 'wuerfel 'ist-ein)
...
(putprop 'a 'rot 'farbe)
```

Putprop ist die Funktion, die eine Eigenschaft in die Propertyliste eines Objektes legt. Das erste Argument 'a ist der Name des Symbols. Unter diesem Namen können wir später auf die Propertyliste zugreifen. Das zweite Argument **'wuerfel** bzw. **'rot** sind die Werte, die die betreffenden Eigenschaften annehmen. Das dritte Argument der putprop-Funktion schließlich (**'ist-ein** bzw. **'farbe**) ist die Eigenschaft, die man dem Objekt zuspricht. Hier zeigt XLISP eine Abweichung vom Common Lisp Standard, in dem erst die Eigenschaft und dann der Wert angeführt werden. Eine Propertyliste ist in XLISP kein normales Symbol, auf das man wie gewohnt zugreifen könnte. Der einzige in XLISP erlaubte Zugriff auf die Propertyliste erfolgt über speziell dafür konzipierte Funktionen: **Putprop** und **Setf** legen eine Eigenschaft auf die entsprechende Propertyliste, **Get** holt den Wert der Eigenschaft einer Propertyliste, **Remprop** entfernt eine Eigenschaft wieder aus der Propertyliste und **Symbol-**

plist gibt die ganze Propertyliste zurück. Es sei bemerkt, daß Putprop und Setf in Bezug auf die Propertylisten gleiche Wirkung zeigen. Die Anwendung der Putprop-Funktion beschränkt sich aber auf die Anwendung auf Propertylisten, während Setf (Abkürzung für Set Field) auch dazu benutzt werden kann, den car einer Liste, den cdr einer Liste, das n.te Element eines Arrays etc. zu setzen. Abb. 2 zeigt den Aufruf und die Wirkung der entsprechenden XLISP-Funktionen auf die Propertyliste des Objektes **'a**, wie sie vom Programm Listing 1 erzeugt wird. Abb. 3 faßt die Syntax der obigen Funktionen in XLISP zusammen.

Der Vollständigkeit halber sei noch bemerkt, daß in Common Lisp und XLISP eine der Propertyliste ähnliche Datenstruktur existiert, die aber in dem obigen Programm nicht verwendet wird: Die Assoziativliste. Hier werden allerdings die Eigenschaft und ihr Wert in einer 2-elementigen Subliste zusammengefaßt. Die Assoziativliste des Objekts 'a würde dann also lauten:

((IST-EIN WUERFEL) (GETRAGEN-VON TISCH) (TRAEGT-DIREKT B) (FARBE ROT))

Der Zugriff auf diese Liste erfolgt mit Hilfe der Assoc-Funktion.

Das LET-Binding

Das LET-Binding gehört in die Gruppe der Kontrollstrukturen, die XLISP zur Verfügung stellt (siehe auch Teil 2 dieser Serie). Um die Wirkung dieser Kontrollstruktur besser zu verstehen, ist es vielleicht sinnvoll zunächst die Begriffe **freie Variable** und **gebundene Variable** zu erläutern. Pascal-Programmierer kennen die Begriffe **globale Variable** und **lokale Variable**. Die beiden Begriffstypen stimmen weitgehend überein, nur daß eben in LISP nicht die formalen Typenbeschränkungen von Pascal existieren. Schauen wir uns als Beispiel für die Wirkung der LET-Bindung die Funktion **pack-auf** mit der zugehörigen Funktion **initialisierung** aus der Blockwelt an: (siehe Beispiel 1).

Die Funktion **initialisierung** enthält keine Parameterliste und keine LET-Bindung. Folglich sind alle in dieser Funktion benutzten Variablen frei. D. h. sie existieren noch, nachdem die

```
> (get 'a 'ist-ein)
WUERFEL
> (symbol-plist 'a)
(FARBE ROT TRAEGT-DIREKT (B) GETRAGEN-VON TISCH GROESSE (2 2 2) POSITION (1 1 0) IST-EIN WUERFEL)
> (remprop 'a 'farbe)
NIL
> (symbol-plist 'a)
(TRAEGT-DIREKT (B) GETRAGEN-VON TISCH GROESSE (2 2 2) POSITION (1 1 0) IST-EIN WUERFEL)
> (setf (get 'a 'farbe) 'rot)
ROT
> (symbol-plist 'a)
(FARBE ROT TRAEGT-DIREKT (B) GETRAGEN-VON TISCH GROESSE (2 2 2) POSITION (1 1 0) IST-EIN WUERFEL)
> (putprop 'a 'gelb 'farbe)
GELB
> (symbol-plist 'a)
(FARBE GELB TRAEGT-DIREKT (B) GETRAGEN-VON TISCH GROESSE (2 2 2) POSITION (1 1 0) IST-EIN WUERFEL)
>
```

Abbildung 2: Protokoll von Abfragen der Propertyliste des Symbols 'a.

```
(putprop <Symbol> <Wert> <Eigenschaft>)

Legt eine Eigenschaft auf die Propertyliste des Symbols. Ist die
Eigenschaft bereits auf der Propertyliste, wird der alte Wert
überschrieben. Gibt den Wert zurück an das Programm.

        <Symbol>: Das Symbol, auf dessen Propertyliste die
                  Eigenschaft gelegt werden soll.
        <Wert>:   Der Wert den die Eigenschaft annehmen soll.
        <Eigenschaft>: Die Eigenschaft, die auf die
                       Propertyliste des Symbols gesetzt werden
                       soll.
________________________________________________________________

(get <Symbol> <Eigenschaft>)

Holt den Wert der Eigenschaft eines Symbols aus der Propertyliste.
Gibt den Wert der Eigenschaft zurück oder NIL, wenn die
Eigenschaft nicht vorhanden ist.

        <Symbol>: Das Symbol, von dem der Wert einer Eigenschaft
                  geholt werden soll.
        <Eigenschaft>: Die Eigenschaft, dessen Wert von der
                       Propertyliste des Symbols geholt werden
                       soll.
________________________________________________________________

(remprop <Symbol> <Eigenschaft>)

Entfernt eine Eigenschaft aus der Propertyliste der Symbols. Gibt
immer NIL zurück.

        <Symbol>: Das Symbol, von dem die Eigenschaft aus der
                  Propertyliste entfernt werden soll.
        <Eigenschaft>: Die Eigenschaft, die entfernt werden
                       soll.
________________________________________________________________

(symbol-plist <Symbol>)

Gibt die gesamte Propertyliste des Symbols zurück.

        <Symbol>: Das Symbol, dessen Propertyliste zurückgegeben
                  werden soll.
________________________________________________________________

(setf <Objekt> <Wert>)

Setzt den Wert eines Objektes. Gibt den Wert zurück. Wirkt wie
putprop, wenn das Objekt die Eigenschaft aus der Propertyliste
eines Symbols ist. Die Anwendung dieser sehr mächtigen Funktion
ist nicht auf Propertylisten beschränkt.

        <Objekt>: Das Objekt, dessen Wert gesetzt werden soll.

                  Bei dem Objekt kann es sich um folgende
                  Datenstrukturen handeln:

                  <Symbol>:       Setzt den Wert eines Symbols.
                  (car <Liste>): Setzt den Wert des ersten
                                  Elementes einer Liste.
                  (cdr <Liste>): Ersetzt die Restliste einer
                                  Liste durch den Wert. Dieser
                                  muß natürlich eine Liste sein.
                  (nth <n> <Liste>):
                                  Setzt den Wert des n.-ten
                                  Elementes der Liste.
                  (aref <Array> <n>):
                                  Setzt den Wert des n.-ten
                                  Elementes eines Arrays.
                  (get <Symbol> <Eigenschaft>):
                                  Setzt den Wert der Eigenschaft
                                  des Symbols.
```

(Fortsetzung auf S. 28)

```
(symbol-value <Symbol>]:
                Setzt den Wert der  Eigenschaft
                eines Symbols.
(symbol-plist <Symbol>):
                Setzt die gesamte Propertyliste
                eines  Symbols.  Wert muß  dann
                natürlich   die   Propertyliste
                sein.
```

(Fortsetzung von S. 27)

Abbildung 3: Syntax der Zugriffsoperationen auf Propertylisten in XLISP.

Funktion verlassen wurde. Damit steht der Aktionsbaum, dessen Wurzeln mit der Funktion **initialisierung** gelegt wurde, für spätere Anfragen zur Verfügung. Im allgemeinen ist dieser Nebeneffekt freier Variablen unerwünscht, weil die Werte freier Variablen in allen Programmteilen verändert werden können, was schnell zu schwer entwanzbaren Programmabstürzen führt. Die Funktion **pack-auf** enthält die Parameterliste mit den gebundenen Variablen **objekt** und **traegt**, stellvertretend für das zu bewegende **objekt** und das Objekt, das es zukünftig tragen soll (**traegt**). In den ersten drei Zeilen des Programms werden mögliche Fehler abgefangen bzw. die Initialisierung des Aktionsbaums und des Plans vorgenommen, d. h. die freien Variablen des Programms werden initialisiert. Das folgende LET-Binding umschließt den wesentlichen Teil der Funktion **pack-auf**. Alle im Bindungsteil (unterstrichen) initialisierten Variablen sind an die LET-Funktion gebunden. Dies sind die Variablen **objekt1** und **traegt1**. Sie werden initialisiert mit den Werten von **objekt** bzw. **traegt**. Gleich darauf folgen zwei Zeilen, indem diese Variablen mit **setq** verändert werden, wenn die entsprechende Bedingung der **cond**-Funktion erfüllt ist. Damit ändert sich aber nichts an der Bindung dieser Variablen! Diese Zeilen ermöglichen es dem Benutzer, ein verschlüsseltes Objekt wie z. B. (**wuerfel rot**), also einen roten Würfel anzugeben. Eine Variante der LET-Bindung ist in der Funktion **traegt-zusaetzlich** zu finden: (Siehe Beispiel 2).

Hier wird in der dritten Zeile des Initialisierungsteils (unterstrichen) die gebundene Variable traegt mit einem Wert initialisiert, der den Wert der ebenfalls gebundenen Variablen **nachfolger1** als bereits vorhanden vorraussetzt. Falls Sie sich noch an Teil 2 dieser Serie erinnern, wissen Sie, daß das normalerweise nicht erlaubt ist, weil die Initialisierung der gebundenen Variablen parallel erfolgt; d. h. es werden erst alle Werte berechnet und dann die Variablen initialisiert. Das würde im Fall der Funktion **traegt-zusaetzlich** zu der Fehlermeldung führen:

Unbound Variable nachfolger 1

Daß die Funktion dennoch funktioniert, dafür sorgt das ★ hinter dem LET! **let★** definiert somit die sequentielle Initialisierung der gebundenen Variablen. Also wird erst die Variable **knoten1** initialisiert, dann die Variable **nachfolger1** und zum Schluß (mit nunmehr initialisierten gebundenen Variablen) **traegt**.

Beispiel 1

```
(defun initialisierung ()
        (setq plan nil)
        (setq knoten 'geschichte)
        (setq nachfolger (gensym))
        (anfuegen nachfolger knoten))
(defun pack-auf (objekt traegt)
        (cond ((equal objekt traegt)
                        (break "Fehler: " (,objekt kann nicht auf sich selbst gepackt werden))))
        (initialisierung)
        (let ((objekt1 objekt)
              (traegt1 traegt))
            (cond ((listp objekt) (setq objekt1 (finde-objekt objekt objektliste))))
            (cond ((listp traegt) (setq traegt1 (finde-objekt traegt objektliste))))
            (putprop nachfolger (list 'pack-auf objekt traegt) 'situation)
            (cond ((equal (get objekt1 'getragen-von) traegt1) (break "Fehler:" (,traegt1 traegt bereits ,objekt1))))
            (pack-hin objekt1 (mach-platz objekt1 traegt1 nachfolger)
            nachfolger) (pp (reverse plan))))
```

Beispiel 2

```
(defun traegt-zusaetzlich (objekt ort &optional nachfolger2)
        (cond ((equal nil nachfolger2)
                (initialisierung) (cond ((listp objekt) (setq objekt (finde-objekt objekt objektliste))))
                (setq nachfolger2 nachfolger)))
    (let★ ((knoten1 nachfolger2)
           (nachfolger1 (gensym))
           (traegt (hole-objekt-unter ort nachfolger1)))
                . . .
```

Manipulation in der Blockwelt

In der Blockwelt stehen nur ganz einfache Manipulationen zur Verfügung. Man kann einen Klotz greifen und ihn irgendwo hinlegen. In Terry Winograds Originalprogramm stehen noch komplizierte Programmteile zur Verfügung, die auch umgangssprachliche Anweisungen wie **pack den roten Klotz auf den blauen** verkraften (siehe hierzu auch den Übersichtsartikel vom gleichen Verfasser). In unserem simplen Programm muß das Objekt entweder bei seinem Symbolnamen genannt werden (also a, b, c, d, e oder f) oder als Liste mit den Eigenschaften **ist-ein** und **farbe**. Um den komplizierten Parser für die Umgangssprache zu vermeiden, wurden die Befehle umgangssprachlich ähnlich aufgebaut. So lautet also der obige Befehl für unsere Blockwelt **(pack-auf 'a 'c)** oder **(pack-auf '(wuerfel rot) '(wuerfel blau))**. Das Programm versucht dann unseren Befehl auszuführen und merkt sich alle auszuführenden Arbeiten in der freien Variable **plan**, die zu Beginn des Programms auf NIL gesetzt wird. Sowohl die greifende Hand als auch die einzelnen Blöcke sind als freie Propertylisten definiert. Alle Funktionen geben den plan in umgekehrter Reihenfolge zurück. Um den Plan besser lesbar zu machen, ist die Funktion **ausgabe** zugefügt worden, die die Liste **plan** in richtiger Reihenfolge ausgibt. Bevor wir uns der Analyse des Programms zuwenden, schauen wir uns die Abarbeitung des obigen Befehls einmal an: (Siehe Beispiel 3).

Abbildung 1: Blockwelt in Listing 1.

Beispiel 3

```
> (ausgabe (pack-auf 'a 'c))
((GREIFE D)
 (BEWEGE-OBJEKT D (RAUM UEBER TISCH FUER D))
 (LASS-LOS D)
 (GREIFE B)
 (BEWEGE-OBJEKT B (RAUM UEBER TISCH FUER B))
 (LASS-LOS B)
 (GREIFE A)
 (BEWEGE-OBJEKT A (RAUM UEBER C FUER A))
 (LASS-LOS A) )
T
>
```

Richtig, wie wir Abb.1 entnehmen, liegt ja auf dem roten Klotz **a** noch ein gelber Klotz **b** und auf dem Würfel **c** liegt noch die Pyramide **d**. Also muß erst die Pyramide **d** von Klotz **c** entfernt werden, dann muß Klotz **b** von Klotz **a** entfernt werden und dann erst kann Würfel **a** auf Würfel **c** gelegt werden. Gut. Nun versuchen wir, den Würfel **b** auf die Kugel zu legen: (Siehe Beispiel 4).

Kein Problem, wie man sieht.

Natürlich ist das Programm stark verbesserungsfähig. Man könnte z. B. noch Koordinaten auf die Propertyliste der Objekte setzen und bei den entsprechenden Manipulationen berücksichtigen. Das wäre ja bei einem realen Roboterarm unbedingt erforderlich. In diesem Programm wird die Koordinatenangabe simuliert über die Mitteilung **(RAUM UEBER __ FUER __)**. Es fehlt auch ein Mechanismus, der erkennt, wenn die Beziehungen nicht eindeutig sind und mit einer Rückfrage reagiert. Also wenn z. B. zwei rote Würfel existieren, müßte das Programm auf den Befehl **(pack-auf '(wuerfel rot) '(schachtel gruen))** mit der Rückfrage reagieren: **Ich weiß nicht welchen roten Würfel du meinst!**

Beispiel 4

```
> (ausgabe (pack-auf 'b 'e))
(BREAK: TRAEGT-ZUSAETZLICH KANN NICHT DAS TRAGEN ARRANGIEREN)
>
```

Stimmt auffallend, auf eine Kugel paßt kein Würfel. Und was ist, wenn die Kugel in die Schachtel soll?

```
> (ausgabe (pack-auf '(kugel gelb) '(schachtel gruen)))
((GREIFE E)
 (BEWEGE-OBJEKT E (RAUM UEBER F FUER E))
 (LASS-LOS E) )
T
>
```

Analyse des Blockweltprogramms

Ich will mich an dieser Stelle nicht zu breit auslassen, weil der interessierte Leser am besten mit dem Programm an seinem XLISP-Interpreter herumspielt. Um die Funktionsweise besser zu verstehen, sollte man alle Funktionen einmal in die Traceliste aufnehmen. Abb. 4 zeigt die entsprechenden Vereinbarungen, die zu Beginn des Programms aufgenommen werden kön-

```
(trace 'initialisierung)
(trace 'pack-auf)
(trace 'finde-objekt)
(trace 'mach-platz)
(trace 'pack-hin)
(trace 'greife)
(trace 'bewege-objekt)
(trace 'bewege-hand)
(trace 'lass-los)
(trace 'werde-los)
(trace 'oben-frei)
(trace 'entferne-traegt)
(trace 'traegt-zusaetzlich)
(trace 'mach-raum)
(trace 'finde-raum)
(trace 'hole-objekt-unter)
(trace 'oben-auf)
(trace 'neu-oben-auf)
(trace 'anfuegen)
```

Abbildung 4: Trace setzen zur Verfolgung der Arbeit im Blockweltprogramm.

nen. Abb. 5 zeigt die entsprechende Ausgabe. Als Hilfe zeigt Abb. 6 eine Übersicht über die Aufgaben der einzelnen Funktionen. Zu erwähnen ist noch das Schlüsselwort **&optional**. Ab diesem Schlüsselwort folgen in der Parameterliste (wer hätte das gedacht?) Parameter, die weggelassen werden können, ohne daß der XLISP-Interpreter einen **TOO FEW ARGUMENTS**-Fehler meldet. Die fehlenden Variablen werden NIL gesetzt, ohne daß die Bindung an die Funktion verlorengeht. In unserem Programm wird diese Eigenschaft genutzt, um z. B. das Greifen eines Blocks alleine durchzuführen. Normalerweise wird ja die Funktion **greife** nur als Folge eines übergeordneten Befehls (wie z. B. **(pack-auf 'a 'c))** ausgeführt. Dann wird aber der Nachfolger des Aktionsbaumes übergeben und das neue Element an diesen Baum angehängt. Man kann unter Benutzung des **&optional**-Schlüsselwortes auch die Hand direkt einen Klotz greifen lassen: **(greife 'a)**. Der fehlende Parameter **nachfolger2** wird selbständig NIL gesetzt und im ersten Teil der Funktion wird die notwendige Initialisierung des Baumes vorgenommen.

```
T
> (pack-auf 'a 'c)
>>> (PACK-AUF (QUOTE A) (QUOTE C))
>>> (INITIALISIERUNG)
>>> (ANFUEGEN NACHFOLGER KNOTEN)
<<< (G1)
<<< (G1)
>>> (PACK-HIN OBJEKT (MACH-PLATZ OBJEKT TRAEGT NACHFOLGER) NACHFOLGER)
>>> (MACH-PLATZ OBJEKT TRAEGT NACHFOLGER)
>>> (ANFUEGEN NACHFOLGER1 KNOTEN1)
<<< (G2)
>>> (FINDE-RAUM OBJEKT TRAEGT NACHFOLGER1)
>>> (ANFUEGEN NACHFOLGER1 KNOTEN1)
<<< (G3)
<<< NIL
>>> (MACH-RAUM OBJEKT TRAEGT NACHFOLGER1)
>>> (ANFUEGEN NACHFOLGER1 KNOTEN1)
<<< (G3 G4)
>>> (WERDE-LOS (CAR DRUEBER) NACHFOLGER1)
>>> (ANFUEGEN NACHFOLGER1 KNOTEN1)
<<< (G5)
>>> (PACK-HIN OBJEKT (FINDE-RAUM OBJEKT (QUOTE TISCH) NACHFOLGER1) NACHFOLGER1)
>>> (FINDE-RAUM OBJEKT (QUOTE TISCH) NACHFOLGER1)
>>> (ANFUEGEN NACHFOLGER1 KNOTEN1)
<<< (G6)
<<< (RAUM UEBER TISCH FUER D)
>>> (ANFUEGEN NACHFOLGER1 KNOTEN1)
<<< (G6 G7)
>>> (GREIFE OBJEKT NACHFOLGER1)
>>> (ANFUEGEN NACHFOLGER1 KNOTEN1)
<<< (G8)
>>> (BEWEGE-HAND (OBEN-AUF OBJEKT NACHFOLGER1) NACHFOLGER1)
>>> (OBEN-AUF OBJEKT NACHFOLGER1)
>>> (ANFUEGEN NACHFOLGER1 KNOTEN1)
<<< (G9)
<<< (OBEN-AUF D)
>>> (ANFUEGEN NACHFOLGER1 KNOTEN1)
<<< (G9 G10)
<<< (OBEN-AUF D)
<<< ((GREIFE D))
>>> (BEWEGE-OBJEKT OBJEKT ORT NACHFOLGER1)
>>> (ANFUEGEN NACHFOLGER1 KNOTEN1)
<<< (G8 G11)
>>> (ENTFERNE-TRAEGT OBJEKT NACHFOLGER1)
>>> (ANFUEGEN NACHFOLGER1 KNOTEN1)
<<< (G12)
<<< NIL
>>> (BEWEGE-HAND (NEU-OBEN-AUF OBJEKT NEUORT NACHFOLGER1) NACHFOLGER1)
>>> (NEU-OBEN-AUF OBJEKT NEUORT NACHFOLGER1)
>>> (ANFUEGEN NACHFOLGER1 KNOTEN1)
<<< (G12 G13)
<<< (NEU-OBEN-AUF D (RAUM UEBER TISCH FUER D))
>>> (ANFUEGEN NACHFOLGER1 KNOTEN1)
<<< (G12 G13 G14)


<<< (NEU-OBEN-AUF D (RAUM UEBER TISCH FUER D))
>>> (TRAEGT-ZUSAETZLICH OBJEKT NEUORT NACHFOLGER1)
>>> (HOLE-OBJEKT-UNTER ORT NACHFOLGER1)
>>> (ANFUEGEN NACHFOLGER1 KNOTEN1)
<<< (G16)
<<< TISCH
>>> (ANFUEGEN NACHFOLGER1 KNOTEN1)
<<< (G12 G13 G14 G15)
<<< TISCH
<<< ((BEWEGE-OBJEKT D (RAUM UEBER TISCH FUER D)) (GREIFE D))
>>> (LASS-LOS OBJEKT NACHFOLGER1)
>>> (ANFUEGEN NACHFOLGER1 KNOTEN1)
<<< (G8 G11 G17)
<<< ((LASS-LOS D) (BEWEGE-OBJEKT D (RAUM UEBER TISCH FUER D)) (GREIFE D))
<<< ((LASS-LOS D) (BEWEGE-OBJEKT D (RAUM UEBER TISCH FUER D)) (GREIFE D))
<<< ((LASS-LOS D) (BEWEGE-OBJEKT D (RAUM UEBER TISCH FUER D)) (GREIFE D))
>>> (FINDE-RAUM OBJEKT TRAEGT NACHFOLGER1)
>>> (ANFUEGEN NACHFOLGER1 KNOTEN1)
<<< (G5 G18)
<<< (RAUM UEBER C FUER A)
<<< (RAUM UEBER C FUER A)
<<< (RAUM UEBER C FUER A)
>>> (ANFUEGEN NACHFOLGER1 KNOTEN1)
<<< (G2 G19)
>>> (GREIFE OBJEKT NACHFOLGER1)
>>> (ANFUEGEN NACHFOLGER1 KNOTEN1)
<<< (G20)
>>> (OBEN-FREI DRUEBER OBJEKT NACHFOLGER1)
>>> (ANFUEGEN NACHFOLGER1 KNOTEN1)
<<< (G21)
>>> (WERDE-LOS (CAR DRUEBER) NACHFOLGER1)
>>> (ANFUEGEN NACHFOLGER1 KNOTEN1)
<<< (G22)
>>> (PACK-HIN OBJEKT (FINDE-RAUM OBJEKT (QUOTE TISCH) NACHFOLGER1) NACHFOLGER1)
>>> (FINDE-RAUM OBJEKT (QUOTE TISCH) NACHFOLGER1)
>>> (ANFUEGEN NACHFOLGER1 KNOTEN1)
<<< (G23)
<<< (RAUM UEBER TISCH FUER B)
>>> (ANFUEGEN NACHFOLGER1 KNOTEN1)
<<< (G23 G24)
>>> (GREIFE OBJEKT NACHFOLGER1)
>>> (ANFUEGEN NACHFOLGER1 KNOTEN1)
<<< (G25)
>>> (BEWEGE-HAND (OBEN-AUF OBJEKT NACHFOLGER1) NACHFOLGER1)
```

(Fortsetzung auf S. 34)

ZWO PUNKT NULL

Zwei der Klassiker zum ATARI ST gibt es jetzt als Neuauflage. Sozusagen als Version 2.0 – völlig überarbeitet und stark erweitert. Zwei Bücher, die bereits in ihrer ersten Auflage mit ihren bestechenden Detailinformationen für Furore sorgten: ATARI ST Intern und ATARI ST 3-D-Grafikprogrammierung. Dabei ist ATARI ST Intern nach wie vor das Informationspaket zum ATARI ST. Selbst große Softwarehäuser aus Amerika wollten bei der Entwicklung ihrer Programme auf dieses Buch nicht verzichten. Wahrscheinlich, weil ATARI ST Intern wirklich alle Informationen zum Aufbau und zur Funktion des ST zusammenfaßt. Ein Werk eben, das bei der professionellen Programmierung unentbehrlich ist – in der Neuauflage noch besser strukturiert und erstmalig mit einer ausführlichen Blitterdokumentation. Ein paar Stichworte aus dem Inhalt: der 68000-Prozessor, der I/O-Controller MFP 68901, der Soundchip YM-2149, das GEMDOS, die BIOS-Funktionen, die XBIOS-Funktionen, der Line-F-Emulator, die Interrupt-Struktur, der VT52-Emulator, die Systemvariablen, der Befehlssatz des 68000-Prozessors und und und. Wen wundert's da noch, daß auch ein kommentiertes BIOS-Listing im Buch enthalten ist.

Brückmann
Englisch
Gerits

ATARI® ST

Intern

EIN DATA BECKER BUCH

Braun

ATARI® ST

3D Grafik Programmierung

EIN DATA BECKER BUCH

Komplettes Know-how bietet auch die Neuauflage zu ATARI ST 3-D-Grafikprogrammierung. Von der nötigen Theorie bis zur Grafikanimation finden Sie hier alles zur 3-D-Grafik: Transformation in der Ebene und im dreidimensionalen Raum, Entfernung der verdeckten Linien, Erzeugen von Rotationskörpern, Definition von mehreren Objekten für Zeichentrickfilm, Licht- und Schatteneffekte. Dabei liegen alle beschriebenen Programme und Routinen sowohl in Assembler wie auch in GFA-BASIC und C vor. Nun hat auch der Einsteiger die Möglichkeit, anhand des reizvollen Themas 3-D-Grafikprogrammierung die Leistungsfähigkeit seines ATARI ST unter Beweis zu stellen.

ATARI ST Intern
Hardcover, ca. 600 Seiten, DM 69,–
erscheint ca. 9/87

ATARI ST 3-D-Grafikprogrammierung
Hardcover, ca. 350 Seiten,
inkl. Diskette, DM 69,–
erscheint ca. 9/87

DATA BECKER
Merowingerstr. 30 · 4000 Düsseldorf · Tel. (0211) 310010

BESTELL-COUPON
Einsenden an: DATA BECKER · Merowingerstr. 30 · 4000 Düsseldorf 1
Bitte senden Sie mir:
☐ per Nachnahme ☐ Verrechnungsscheck liegt bei
Name
Straße
Ort
ST 107

Der Aktionsbaum

In der Initialisierungsfunktion wird die Wurzel eines Aktionsbaumes gelegt (**knoten** genannt), der es ermöglicht, nach Ablauf einer Aktion Fragen nach den Wie, Weshalb und Warum zu beantworten. Dazu wird jede Funktion mit dem optionalen Parameter **nachfolger** aufgerufen. Dieser wird an die Variable **knoten1** lokal gebunden, gleichzeitig wird ein neuer Knoten (**nachfolger1** genannt) mit Hilfe der Funktion **gensym** erzeugt. In den jeweils aktuellen Knoten wird dann die in der jeweiligen Funktion ablaufende Aktion abgespeichert. Abb. 8 zeigt das Listing der Funktion, mit deren Hilfe man den Aktionsbaum auslisten kann.

Die zugehörige Ausgabe für die Aktion **(pack-auf 'a 'c)** findet man in Abb. 7. Die Verbindungslinien wurden allerdings mit Hand eingetragen.

In diesem Aktionsbaum suchen die Funktionen **sag-wie, sag-weshalb** und **sag-warum** nach den entsprechenden Antworten. Die Funktion **sag-wie** soll die Antwort auf die Methode finden, wie das gesteckte Ziel erreicht werden kann. Sie muß also im Aktionsbaum nach dem Ziel als Schlüssel suchen und die Nachfolger dieses Schlüssels ausgeben. Hierzu wird einfach rekursiv der Aktionsbaum abgetastet und entweder bei Auffinden des Schlüssels die Nachfolger ausgegeben oder bei Fehlen des Schlüssels NIL ausgegeben. Zur Ausgabe der Nachfolger noch ein Wort: Hier findet die Funktion **mapcar** Anwendung. Als erstes Argument verlangt diese Funktion eine Funktionsdefinition. Als zweites Argument muß eine Liste pro Argument der Funktionsdefinition folgen. Die Funktion **mapcar** holt sich dann aus jeder Liste das erste Element und übergibt es an die Parameterliste der Funktionsdefinition. Der Vorgang wird dann mit dem cdr der Listen wiederholt, bis die Listen leer sind. Die Antwort auf die Frage **weshalb** wird von der Funktion **sag-weshalb** ausgegeben. Hierzu muß zunächst wieder der Schlüssel gesucht werden und anschließend die Vorgängerliste ausgegeben werden. Die Frage nach dem Warum ist besonders einfach zu beantworten. Hierzu ist lediglich die Wurzel des Baumes auszugeben.

(Fortsetzung von S. 32)

```
>>> (OBEN-AUF OBJEKT NACHFOLGER1)
>>> (ANFUEGEN NACHFOLGER1 KNOTEN1)
<<< (G26)
<<< (OBEN-AUF B)
>>> (ANFUEGEN NACHFOLGER1 KNOTEN1)
<<< (G26 G27)
<<< (OBEN-AUF B)
<<< ((GREIFE B) (LASS-LOS D) (BEWEGE-OBJEKT D (RAUM UEBER TISCH FUER D)) (GREIFE D))
>>> (BEWEGE-OBJEKT OBJEKT ORT NACHFOLGER1)
>>> (ANFUEGEN NACHFOLGER1 KNOTEN1)
<<< (G25 G28)
>>> (ENTFERNE-TRAEGT OBJEKT NACHFOLGER1)
>>> (ANFUEGEN NACHFOLGER1 KNOTEN1)
<<< (G29)
<<< NIL
>>> (BEWEGE-HAND (NEU-OBEN-AUF OBJEKT NEUORT NACHFOLGER1) NACHFOLGER1)
>>> (NEU-OBEN-AUF OBJEKT NEUORT NACHFOLGER1)
>>> (ANFUEGEN NACHFOLGER1 KNOTEN1)
<<< (G29 G30)
<<< (NEU-OBEN-AUF B (RAUM UEBER TISCH FUER B))
>>> (ANFUEGEN NACHFOLGER1 KNOTEN1)
<<< (G29 G30 G31)
<<< (NEU-OBEN-AUF B (RAUM UEBER TISCH FUER B))
>>> (TRAEGT-ZUSAETZLICH OBJEKT NEUORT NACHFOLGER1)
>>> (HOLE-OBJEKT-UNTER ORT NACHFOLGER1)
>>> (ANFUEGEN NACHFOLGER1 KNOTEN1)
<<< (G33)
<<< TISCH
>>> (ANFUEGEN NACHFOLGER1 KNOTEN1)
<<< (G29 G30 G31 G32)
<<< TISCH
<<< ((BEWEGE-OBJEKT B (RAUM UEBER TISCH FUER B)) (GREIFE B) (LASS-LOS D) (BEWEGE-OBJEKT D (RAUM UEBER TISCH FUER D))
(GREIFE D))
>>> (LASS-LOS OBJEKT NACHFOLGER1)
>>> (ANFUEGEN NACHFOLGER1 KNOTEN1)
<<< (G25 G28 G34)
<<< ((LASS-LOS B) (BEWEGE-OBJEKT B (RAUM UEBER TISCH FUER B)) (GREIFE B) (LASS-LOS D) (BEWEGE-OBJEKT D (RAUM UEBER TI
SCH FUER D)) (GREIFE D))
<<< ((LASS-LOS B) (BEWEGE-OBJEKT B (RAUM UEBER TISCH FUER B)) (GREIFE B) (LASS-LOS D) (BEWEGE-OBJEKT D (RAUM UEBER TI
SCH FUER D)) (GREIFE D))
<<< ((LASS-LOS B) (BEWEGE-OBJEKT B (RAUM UEBER TISCH FUER B)) (GREIFE B) (LASS-LOS D) (BEWEGE-OBJEKT D (RAUM UEBER TI
SCH FUER D)) (GREIFE D))
<<< NIL
>>> (BEWEGE-HAND (OBEN-AUF OBJEKT NACHFOLGER1) NACHFOLGER1)
>>> (OBEN-AUF OBJEKT NACHFOLGER1)
>>> (ANFUEGEN NACHFOLGER1 KNOTEN1)
<<< (G21 G35)
<<< (OBEN-AUF A)
>>> (ANFUEGEN NACHFOLGER1 KNOTEN1)
<<< (G21 G35 G36)
<<< (OBEN-AUF A)
<<< ((GREIFE A) (LASS-LOS B) (BEWEGE-OBJEKT B (RAUM UEBER TISCH FUER B)) (GREIFE B) (LASS-LOS D) (BEWEGE-OBJEKT D (RA
UM UEBER TISCH FUER D)) (GREIFE D))
>>> (BEWEGE-OBJEKT OBJEKT ORT NACHFOLGER1)
>>> (ANFUEGEN NACHFOLGER1 KNOTEN1)
<<< (G28 G37)
>>> (ENTFERNE-TRAEGT OBJEKT NACHFOLGER1)
>>> (ANFUEGEN NACHFOLGER1 KNOTEN1)
<<< (G38)
<<< NIL
>>> (BEWEGE-HAND (NEU-OBEN-AUF OBJEKT NEUORT NACHFOLGER1) NACHFOLGER1)
>>> (NEU-OBEN-AUF OBJEKT NEUORT NACHFOLGER1)
>>> (ANFUEGEN NACHFOLGER1 KNOTEN1)
<<< (G38 G39)
<<< (NEU-OBEN-AUF A (RAUM UEBER C FUER A))
>>> (ANFUEGEN NACHFOLGER1 KNOTEN1)
<<< (G38 G39 G40)
<<< (NEU-OBEN-AUF A (RAUM UEBER C FUER A))
>>> (TRAEGT-ZUSAETZLICH OBJEKT NEUORT NACHFOLGER1)
>>> (HOLE-OBJEKT-UNTER ORT NACHFOLGER1)
>>> (ANFUEGEN NACHFOLGER1 KNOTEN1)
<<< (G42)
<<< C
>>> (ANFUEGEN NACHFOLGER1 KNOTEN1)
<<< (G38 G39 G40 G41)
<<< C
<<< ((BEWEGE-OBJEKT A (RAUM UEBER C FUER A)) (GREIFE A) (LASS-LOS B) (BEWEGE-OBJEKT B (RAUM UEBER TISCH FUER B)) (G
IFE B) (LASS-LOS D) (BEWEGE-OBJEKT D (RAUM
 UEBER TISCH FUER D)) (GREIFE D))
>>> (LASS-LOS OBJEKT NACHFOLGER1)
>>> (ANFUEGEN NACHFOLGER1 KNOTEN1)
<<< (G28 G37 G43)
<<< ((LASS-LOS A) (BEWEGE-OBJEKT A (RAUM UEBER C FUER A)) (GREIFE A) (LASS-LOS B) (BEWEGE-OBJEKT B (RAUM UEBER TIS
FUER B)) (GREIFE B) (LASS-LOS D) (BEWEGE-O
 BJEKT D (RAUM UEBER TISCH FUER D)) (GREIFE D))
<<< ((LASS-LOS A) (BEWEGE-OBJEKT A (RAUM UEBER C FUER A)) (GREIFE A) (LASS-LOS B) (BEWEGE-OBJEKT B (RAUM UEBER TIS
FUER B)) (GREIFE B) (LASS-LOS D) (BEWEGE-O
 BJEKT D (RAUM UEBER TISCH FUER D)) (GREIFE D))
<<< ((LASS-LOS A) (BEWEGE-OBJEKT A (RAUM UEBER C FUER A)) (GREIFE A) (LASS-LOS B) (BEWEGE-OBJEKT B (RAUM UEBER TIS
FUER B)) (GREIFE B) (LASS-LOS D) (BEWEGE-O
 BJEKT D (RAUM UEBER TISCH FUER D)) (GREIFE D))
((LASS-LOS A) (BEWEGE-OBJEKT A (RAUM UEBER C FUER A)) (GREIFE A) (LASS-LOS B) (BEWEGE-OBJEKT B (RAUM UEBER TISCH FU
B)) (GREIFE B) (LASS-LOS D) (BEWEGE-OBJEK
 T D (RAUM UEBER TISCH FUER D)) (GREIFE D))
) (ausgabe plan)
((GREIFE D)
 (BEWEGE-OBJEKT D (RAUM UEBER TISCH FUER D))
 (LASS-LOS D)
 (GREIFE B)
```

(Fortsetzung auf S. 36)

Wenn Ihnen die Zeit im Nacken sitzt,

sollten Sie sich für OMIKRON.BASIC entscheiden.

Wie Sie der Fachpresse entnehmen können, ist unser OMIKRON.BASIC der schnellste Basic-Interpreter, den es für den ATARI ST gibt.*

* Siehe XEST Nr. 011 Seite 73 und ST COMPUTER Nr. 9/87 Seite 64.

ARITHMETIK: Rechengenauigkeit bis 19 Stellen bei allen Funktionen ● Rechenbereich bis 5.11 E±4931

GEM: Komplette GEM-Library ● Sämtliche AES- und VDI-Funktionen direkt mit Namen verfügbar ● Eigene BITBLIT-Routine

EXTRAS: Masken-INPUT ● SORT-Befehl sortiert beliebige Felder, auch mit Umlauten ● Matrizenbefehle

STRUKTUR: Prozeduren und mehrzeilige Funktionen mit Übergabe-und Rückgabe-Parametern und lokalen Variablen ● REPEAT ... UNTIL, WHILE ... WEND, mehrzeiliges IF ... THEN ... ELSE ... ENDIF ● Labels bei GOTO, GOSUB, ON GOTO und ON GOSUB

EDITOR: Mit oder ohne Zeilennummern (umschaltbar) ● Drei Schriftgrößen bis 57x128 Zeichen ● Frei definierbare Teach-In-Funktionstasten

GESCHWINDIGKEIT: FIT-Code (FIT = Fast Interpreting Technique) ● Wir kennen keine schnelleren 68000-Fließkomma-Routinen ● Volle Integer-Arithmetik ● Eigene Disk-Routinen für beschleunigten Dateizugriff

KOMPATIBILITÄT: Für alle ATARI ST (auch MEGA ST) ● 99% MBASIC-kompatibel Editor findet Inkompatibilitäten, dadurch einfachste Anpassung

● Diskettenversion mit Demodiskette und Handbuch **DM 179,–**

● Modulversion mit Demodiskette und Handbuch **DM 229,–**

COMPILER

Für alle, denen die Zeit nicht nur im Nacken sitzt, sondern bereits auf den Nägeln brennt, entwickelten wir den OMIKRON.BASIC-COMPILER.

Er erlaubt es Ihnen, bequem in Basic interpretativ zu programmieren, um nach der Compilierung Geschwindigkeiten zu erreichen, die bisher C-Compilern vorbehalten waren. Der Aufbruch in neue Dimensionen der Basic-Programmierung hat begonnen.

● Diskette mit Anleitung **DM 179,–**

OMIKRON.

... denn das Beste ist für Ihren ATARI ST gerade gut genug!

OMIKRON.SOFTWARE
Erlachstraße 15 · D-7534 Birkenfeld · ☎ (0 70 82) 53 86

Österreich: Ueberreuter Media · Alser Straße 24 · A-1091 Wien
Schweiz: Microtron · Postfach 40 · CH-4542 Pieterlen

OMIKRON. BASIC

OMIKRON. COMPILER für den ATARI ST

Ausblick

Die Serie geht langsam ihrem Ende zu. Ich möchte Ihnen aber in den verbleibenden beiden Ausgaben zwei Verfahren zur Wissensspeicherung erläutern: Die Rahmentechnologie und das Matrixkonzept.

Dr. Sarnow

Literatur:

[1] Sarnow, K.: Künstliche Intelligenz. Einführung. ST-Computer, Heft 11/86.

[2] Winston, P.H. und B. Horn: LISP. Addison-Wesley Publishing, Reading Massachusetts, 1984.

[3] Winograd, Terry: Understanding Natural Language. New York 1972.

(Fortsetzung von S. 34)

```
(BEWEGE-OBJEKT B (RAUM UEBER TISCH FUER B))
(LASS-LOS B)
(GREIFE A)
(BEWEGE-OBJEKT A (RAUM UEBER C FUER A))
(LASS-LOS A) )
T
> (exit)
```

Abbildung 5: Trace der Blockwelt beim Abarbeiten von (pack-auf 'a 'c).

Name der Funktion	Beschreibung der Aufgabe
PACK-AUF	Legt ein Objekt auf ein anderes ab. Die zentrale Funktion des Blockweltprogramms. Alle anderen Funktionen sind mehr oder weniger Hilfsfunktionen dieser Funktion.
PACK-HIN	Unterscheidet sich von PACK-AUF, indem das Objekt an eine freie Stelle gelegt wird.
MACH-PLATZ	Beseitigt alle Objekte, die der durchzuführenden Aktion im Wege sind.
FINDE-RAUM	Versucht an der gewünschten Stelle einen leeren Raum zu finden.
MACH-RAUM	Räumt die gewünschte Stelle leer.
WERDE-LOS	Legt die Objekte, die der Aktion im Wege sind auf dem Tisch ab.
HOLE-OBJEKT-UNTER	Ermittelt die Liste der Objekte, die sich unter einem Objekt befinden.

Abbildung 6: Übersicht über die wichtigsten Funktionen der Blockwelt. Die meisten Namen sind selbsterklärend.

```
T
> (baum (list knoten) 1)
      |—(FINDE-RAUM A C)
    |-(MACH-PLATZ A C)
    | |       |—(FINDE-RAUM D TISCH)
    | |    |—(WERDE-LOS D)
    | |    | |      |—(OBEN-AUF D)
    | |    | |  |—(GREIFE D)
    | |    | |  |  |—(BEWEGE-HAND (OBEN-AUF D))
    | |    | |—(PACK-HIN D (RAUM UEBER TISCH FUER D))
    | |    |    |  |—(ENTFERNE-TRAEGT D)
    | |    |    |—(BEWEGE-OBJEKT D (RAUM UEBER TISCH FUER D))
    | |    |       |—(NEU-OBEN-AUF D (RAUM UEBER TISCH FUER D))
    | |—(MACH-RAUM A C)
    |      |—(FINDE-RAUM A C)
    (PACK-AUF A C)
    |             |—(FINDE-RAUM B TISCH)
    |         |—(WERDE-LOS B)
    |         | |      |—(OBEN-AUF B)
    |         | |  |—(GREIFE B)
    |         | |  |  |—(BEWEGE-HAND (OBEN-AUF B))
    |         | |—(PACK-HIN B (RAUM UEBER TISCH FUER B))
    |         |    |  |—(ENTFERNE-TRAEGT B)
    |         |    |—(BEWEGE-OBJEKT B (RAUM UEBER TISCH FUER B))
    |         |       |—(NEU-OBEN-AUF B (RAUM UEBER TISCH FUER B))
    |      |—(OBEN-FREI (B) A)
    |  |—(GREIFE A)
    |  |  |—(OBEN-AUF A)
    |—(PACK-HIN A (RAUM UEBER C FUER A))
       |  |—(ENTFERNE-TRAEGT A)
       |—(BEWEGE-OBJEKT A (RAUM UEBER C FUER A))
          |—(NEU-OBEN-AUF A (RAUM UEBER C FUER A))
NIL
T
> (exit)
```

Abbildung 7

```
(defun baum (knoten stufe)
      (cond ((equal nil knoten) T)
               (t (let ((knoten1 (get (car knoten) 'nachfolger)))
                   (baum knoten1 (+ stufe 1))
                   (do ((i 1 (+ i 1)))
                        ((equal i stufe))
                        (princ "   "))
                   (print (get (car knoten) 'situation))
                     (baum (cdr knoten1) (+ 1 stufe)) ))))
```

Abbildung 8: Funktion zum Ausdrucken des Aktionsbaumes.

ATARI ST Bibliothek

Ob frischgebackener ST-Besitzer oder ambitionierter 68000er-Programmierer – wer seinen ATARI ST effizient und professionell einsetzen will, der braucht hochrangige Informationen von hochkarätigen Autoren. Informationen, die genau auf seine Bedürfnisse und Anwendungsgebiete zugeschnitten sind. Informationen, die Sie in unserer ST-Bibliothek finden können. Eine kleine Auswahl stellen wir Ihnen hier vor.

Lüers · Stein
ATARI ST für Einsteiger
EIN DATA BECKER BUCH

Das erste Buch zum ATARI ST ist für viele das wichtigste. ATARI ST für Einsteiger ist eine leichtverständliche Einführung in Handhabung, Einsatz und Programmierung Ihres STs: Von der Tastatur und der Maus hin zum Desktop bis zum ersten BASIC-Befehl und schließlich zum ganzen Programm. Die Autoren zeigen Ihnen den Weg zum Erfolg mit Ihrem neuen Rechner.
ATARI ST für Einsteiger
262 Seiten, DM 29,–

Brückmann · Englisch · Gerits · Walkowiak
ATARI ST Tips & Tricks
EIN DATA BECKER BUCH

Tips und Tricks zum ATARI ST – das Buch, das voll guter Ideen steckt. Sie suchen einen Druckerspooler? In diesem Buch finden Sie ihn. Sie brauchen eine Farb-Hardcopy-Routine? Bitte schön. Automatisches Starten von TOS-Anwendungen? Nichts leichter als das. Eine RAM-Disk anlegen? Nur zu. Mit vielen, neuen Routinen, die problemlos in eigene Programme eingebunden werden können.
ATARI ST Tips & Tricks
Hardcover, 352 Seiten, DM 49,–

Grohmann · Seidler · Slibar
Das Maschinensprachebuch zum ATARI ST
EIN DATA BECKER BUCH

68000-Assembler – der Schlüssel zur Entwicklung der leistungsfähigsten Programme auf dem ATARI ST. Dieses Buch zeigt Ihnen, wie Sie Ihren ST mit Maschinensprache voll ausnutzen können: Verwendung von Systemroutinen, Bitmanipulationen, Rekursion, Stacks, Einbinden von Assemblerroutinen in Hochsprachen – vom ersten Schritt bis hin zum echten Einsatz von Maschinensprache.
Das Maschinensprachebuch zum ATARI ST
334 Seiten, DM 39,–

Plenge
Das SUPER GRAFIK Buch zum ATARI ST
MIT DISKETTE
EIN DATA BECKER BUCH

Die Grafikfähigkeit des STs gezielt für eigene Anwendungen einsetzen – dieses Buch zeigt Ihnen, wie es geht. Angefangen von den Grundlagen des VDI, GEM, AES und TOS bis hin zu speziellen Problemlösungen wie Programmierung des Rasterinterrupts oder einer flackerfreien Animation finden Sie hier alles zum Thema Grafik auf dem ST. Mit zahlreichen Utilities in BASIC, C und Assembler.
Das Supergrafikbuch zum ATARI ST
Hardcover, 838 Seiten,
inkl. Diskette, DM 69,–

Liesert · Linden
ATARI ST Einführung in CAD
MIT DISKETTE
EIN DATA BECKER BUCH

CAD ist mehr als „nur" Computergrafik. Neben den speziellen Programmiertechniken und den typischen CAD-Prozeduren braucht man noch solide Grundkenntnisse zum Aufbau eines CAD-Systems. Wissen, das in diesem Buch kompakt und leichtverständlich zusammengefaßt wurde. Zudem können Sie anhand der einzelnen beschriebenen Module und mit GFA-BASIC 2.0 leicht ein komplettes CAD-Programm erstellen.
ATARI ST – Einführung in CAD
Hardcover, 289 Seiten,
inkl. Diskette, DM 69,–

Schaun
ATARI ST C für Einsteiger
EIN DATA BECKER BUCH

Sie suchen einen praxisorientierten, einfachen Einstieg in C? Hier ist er: C für Einsteiger. Doch beschränkt sich dieses Buch nicht nur auf die grundsätzliche Bedienung und Struktur von C, sondern vermittelt auch echtes Profi-Wissen. So z.B. zur GEM- und Fensterprogrammierung. Mit den zahlreichen Tips und Tricks formt dieses Buch aus jedem Einsteiger einen Profi.
ATARI ST – C für Einsteiger
393 Seiten, DM 39,–

Abraham · Englisch · Günther · Szczepanowski
ATARI ST GEM
EIN DATA BECKER BUCH

ATARI ST GEM – ein schlichter, präziser Titel für ein Buch, das alles bietet, was es zu GEM zu sagen gibt: Systemaufrufe aus GFA-BASIC, C und Assembler. Erstellung eigener GEM-Bindings, Aufbau der Resourcen, Programmierung von Slider-Objekten, Aufbau eines eigenen Desktop und vor allen Dingen: ein komplett kommentiertes VDI-Listing sowie ein kommmentiertes Listing ausgewählter AES-Funktionen.
ATARI ST GEM
Hardcover, 691 Seiten, DM 69,–

Der DATA BECKER Führer
1st Word
1st Word Plus
1st Mailmaster
1st Mail
1st Lektor
1st Spooler
Alles auf einen Blick

Arbeiten Sie mit einem 1st-Word-Programm? Dann sollten Sie auf diesen DATA BECKER Führer nicht verzichten. Denn hier finden Sie die Antworten zu allen brennenden Fragen: Wie installiert man einen Drucker? Lassen sich Grafiken einbinden? Wie lassen sich die Programme untereinander verknüpfen? Egal worüber Sie mehr wissen wollen: Ein Blick in diesen DATA BECKER Führer genügt.
Der DATA BECKER Führer
zu 1st Word
196 Seiten, DM 24,80

DATA BECKER

Merowingerstr. 30 · 4000 Düsseldorf · Tel. (0211) 310010

BESTELL-COUPON
Einsenden an: DATA BECKER · Merowingerstr. 30 · 4000 Düsseldorf 1
Bitte senden Sie mir:

zzgl. DM 5,- Versandkosten unabhängig von der bestellten Stückzahl
□ per Nachnahme □ Verrechnungsscheck liegt bei

Name
Straße
Ort

```
(putprop 'hand NIL 'greifend)          ;Die Hand greift nichts

(putprop 'a 'wuerfel 'ist-ein)         ;Es folgt die Definition
(putprop 'a 'tisch 'getragen-von)      ;der Objekte der Blockwelt.
(putprop 'a '(b) 'traegt-direkt)       ;Objekt a ist also ein roter
(putprop 'a 'rot 'farbe)               ;Würfel der auf dem Tisch
                                       ;liegt und Objekt b trägt.
(putprop 'b 'wuerfel 'ist-ein)
(putprop 'b 'a 'getragen-von)
(putprop 'b '() 'traegt-direkt)
(putprop 'b 'gelb 'farbe)

(putprop 'c 'wuerfel 'ist-ein)
(putprop 'c 'tisch 'getragen-von)
(putprop 'c '(d) 'traegt-direkt)
(putprop 'c 'blau 'farbe)

(putprop 'd 'pyramide 'ist-ein)
(putprop 'd 'c 'getragen-von)
(putprop 'd '() 'traegt-direkt)
(putprop 'd 'gruen 'farbe)

(putprop 'e 'kugel 'ist-ein)
(putprop 'e 'tisch 'getragen-von)
(putprop 'e '() 'traegt-direkt)
(putprop 'e 'gelb 'farbe)


(putprop 'f 'schachtel 'ist-ein)
(putprop 'f 'tisch 'getragen-von)
(putprop 'f '() 'traegt-direkt)
(putprop 'f 'gruen 'farbe)

(setq objektliste '(a b c d e f))

(defun initialisierung ()
       (setq plan nil)
       (setq knoten 'geschichte)
       (setq nachfolger (gensym))
       (anfuegen nachfolger knoten))

(defun pack-auf (objekt traegt)
       (cond ((equal objekt traegt)
                    (break "Fehler: " '(,objekt kann nicht auf sich selbst gepackt werden))))
       (initialisierung)
       (cond ((listp objekt) (setq objekt (finde-objekt objekt objektliste))))
       (cond ((listp traegt) (setq traegt (finde-objekt traegt objektliste))))
       (putprop nachfolger (list 'pack-auf objekt traegt) 'situation)
       (cond ((equal (get objekt 'getragen-von) traegt) (break "Fehler:" '(,traegt traegt bereits ,objekt))))
       (pack-hin objekt (mach-platz objekt traegt nachfolger) nachfolger))

(defun finde-objekt (objekt objektliste)
       (cond ((equal objektliste nil) (break "Objekt nicht in Objektliste:" objekt)))


       (cond ((equal (get (car objektliste) 'ist-ein) (car objekt))
                            (cond ((equal (get (car objektliste) 'farbe) (cadr objekt))
                                          (car objektliste))
                                     (t (finde-objekt objekt (cdr objektliste)))))
                (t (finde-objekt objekt (cdr objektliste)))))


(defun mach-platz (objekt traegt &optional nachfolger2)
       (cond ((equal objekt traegt)
                    (break "Fehler: " '(,objekt kann nicht fuer sich selbst platz machen))))
       (cond ((equal nil nachfolger2)
                            (initialisierung)
                                     (cond ((listp objekt) (setq objekt (finde-objekt objekt objektliste))))
                                     (cond ((listp traegt) (setq traegt (finde-objekt traegt objektliste))))
                                           (setq nachfolger2 nachfolger)))
    (let ((knoten1 nachfolger2)
          (nachfolger1 (gensym)))
         (anfuegen nachfolger1 knoten1)
         (putprop nachfolger1 (list 'mach-platz objekt traegt) 'situation)
         (cond ((finde-raum objekt traegt nachfolger1))
               ((mach-raum objekt traegt nachfolger1)))))

(defun pack-hin (objekt ort &optional nachfolger2)
       (cond ((equal nil nachfolger2)
                            (initialisierung)
                                     (cond ((listp objekt) (setq objekt (finde-objekt objekt objektliste))))
                                           (setq nachfolger2 nachfolger)))
    (let ((knoten1 nachfolger2)
          (nachfolger1 (gensym)))
         (anfuegen nachfolger1 knoten1)
         (putprop nachfolger1 (list 'pack-hin objekt ort) 'situation)
         (greife objekt nachfolger1)
         (bewege-objekt objekt ort nachfolger1)
         (lass-los objekt nachfolger1)))

(defun greife (objekt &optional nachfolger2)
       (cond ((not (equal (get 'hand 'greifend) nil))
                            (break "Fehler: Hand greift bereits " (get 'hand 'greifend))))
       (cond ((equal nil nachfolger2)
                            (initialisierung)
                                     (cond ((listp objekt) (setq objekt (finde-objekt objekt objektliste))))
                                           (setq nachfolger2 nachfolger)))
    (let ((darueber (get 'hand 'greifend))
          (drueber (get objekt 'traegt-direkt))
          (knoten1 nachfolger2)
          (nachfolger1 (gensym)))
         (anfuegen nachfolger1 knoten1)
         (putprop nachfolger1 (list 'greife objekt) 'situation)
         (cond (darueber (werde-los darueber nachfolger1)))
         (cond (drueber (oben-frei drueber objekt nachfolger1)))
         (bewege-hand (oben-auf objekt nachfolger1) nachfolger1)
         (setf (get 'hand 'greifend) objekt)
         (setq plan (cons (list 'greife objekt) plan))))

(defun bewege-objekt (objekt neuort &optional nachfolger2)

       (cond ((equal nil nachfolger2)
                            (initialisierung)
                                     (cond ((listp objekt) (setq objekt (finde-objekt objekt objektliste))))
                                           (setq nachfolger2 nachfolger)))
    (let ((knoten1 nachfolger2)
          (nachfolger1 (gensym)))
         (anfuegen nachfolger1 knoten1)
         (putprop nachfolger1 (list 'bewege-objekt objekt neuort) 'situation)
         (entferne-traegt objekt nachfolger1)
         (bewege-hand (neu-oben-auf objekt neuort nachfolger1) nachfolger1)
         (setf (get objekt 'position) neuort)
         (traegt-zusaetzlich objekt neuort nachfolger1)
         (setq plan (cons (list 'bewege-objekt objekt neuort) plan))))

(defun bewege-hand (position &optional nachfolger2)
       (cond ((equal nil nachfolger2)
                            (initialisierung)
                                     (cond ((listp objekt) (setq objekt (finde-objekt objekt objektliste))))
                                           (setq nachfolger2 nachfolger)))
    (let ((knoten1 nachfolger2)
          (nachfolger1 (gensym)))
         (anfuegen nachfolger1 knoten1)
         (putprop nachfolger1 (list 'bewege-hand position) 'situation)
         (setf (get 'hand 'position) position)))
```

```
(defun lass-los (objekt &optional nachfolger2)
       (cond ((equal (get 'hand 'greifend) nil)
                     (break "Fehler: " '(hand greift nichts))))
       (cond ((equal nil nachfolger2)
                        (initialisierung)
                     (cond ((listp objekt) (setq objekt (finde-objekt objekt objektliste))))
                        (setq nachfolger2 nachfolger)))
    (let ((knoten1 nachfolger2)
          (nachfolger1 (gensym)))
         (anfuegen nachfolger1 knoten1)
         (putprop nachfolger1 (list 'lass-los objekt) 'situation)
         (cond ((not (get objekt 'getragen-von)) nil)
               (T (setf (get 'hand 'greifend) nil)
                  (setq plan (cons (list 'lass-los objekt) plan))))))

(defun werde-los (objekt &optional nachfolger2)
       (cond ((equal nil nachfolger2)
                        (initialisierung)
                     (cond ((listp objekt) (setq objekt (finde-objekt objekt objektliste))))
                        (setq nachfolger2 nachfolger)))
    (let ((knoten1 nachfolger2)
          (nachfolger1 (gensym)))
         (anfuegen nachfolger1 knoten1)
         (putprop nachfolger1 (list 'werde-los objekt) 'situation)
         (pack-hin objekt (finde-raum objekt 'tisch nachfolger1) nachfolger1)))

(defun oben-frei (drueber objekt &optional nachfolger2)
       (cond ((equal nil nachfolger2)
                        (initialisierung)

                    (cond ((listp objekt) (setq objekt (finde-objekt objekt objektliste))))
                        (setq nachfolger2 nachfolger)))
    (let ((knoten1 nachfolger2)
          (nachfolger1 (gensym)))
         (anfuegen nachfolger1 knoten1)
         (putprop nachfolger1 (list 'oben-frei drueber objekt) 'situation)
         (do ((drueber drueber (cdr drueber)))
             ((null drueber))
             (werde-los (car drueber) nachfolger1))))

(defun entferne-traegt (objekt &optional nachfolger2)
       (cond ((equal nil nachfolger2)
                        (initialisierung)
                    (cond ((listp objekt) (setq objekt (finde-objekt objekt objektliste))))
                        (setq nachfolger2 nachfolger)))
  (let ((traegt (get objekt 'getragen-von))
        (knoten1 nachfolger2)
        (nachfolger1 (gensym)))
       (anfuegen nachfolger1 knoten1)
       (putprop nachfolger1 (list 'entferne-traegt objekt) 'situation)
       (setf (get traegt 'traegt-direkt)
       (remove objekt (get traegt 'traegt-direkt)))
       (setf (get objekt 'getragen-von) nil)))

(defun traegt-zusaetzlich (objekt ort &optional nachfolger2)
       (cond ((equal nil nachfolger2)
                        (initialisierung)
                    (cond ((listp objekt) (setq objekt (finde-objekt objekt objektliste))))
                        (setq nachfolger2 nachfolger)))
  (let* ((knoten1 nachfolger2)
         (nachfolger1 (gensym))
         (traegt (hole-objekt-unter ort nachfolger1)))
        (anfuegen nachfolger1 knoten1)
        (putprop nachfolger1 (list 'traegt-zusaetzlich objekt ort) 'situation)
        (cond  ((or (equal traegt 'tisch)
                    (equal (get traegt 'ist-ein) 'schachtel)
                    (equal (get traegt 'ist-ein) 'wuerfel))
                       (setf (get traegt 'traegt-direkt)
                         (cons objekt (get traegt 'traegt-direkt)))
                       (setf (get objekt 'getragen-von) traegt))
               (t (break '(traegt-zusaetzlich kann nicht das tragen arrangieren))))))

(defun mach-raum (objekt traegt &optional nachfolger2)
       (cond ((equal objekt traegt)
                     (break "Fehler: " '(,objekt kann nicht fuer sich selbst raum machen))))
       (cond ((equal nil nachfolger2)
                        (initialisierung)
                     (cond ((listp objekt) (setq objekt (finde-objekt objekt objektliste))))
                        (setq nachfolger2 nachfolger)))
    (let ((knoten1 nachfolger2)
          (nachfolger1 (gensym)))
         (anfuegen nachfolger1 knoten1)
         (putprop nachfolger1 (list 'mach-raum objekt traegt) 'situation)
         (do ((drueber (get traegt 'traegt-direkt) (cdr drueber))


             (ort nil))
                 (ort ort)
             (werde-los (car drueber) nachfolger1)
             (setq ort (finde-raum objekt traegt nachfolger1)))))

(defun finde-raum (objekt traegt &optional nachfolger2)
       (cond ((equal objekt traegt)
                     (break "Fehler: " '(,objekt kann nicht fuer sich selbst raum finden))))
       (cond ((equal nil nachfolger2)
                        (initialisierung)
                     (cond ((listp objekt) (setq objekt (finde-objekt objekt objektliste))))
                        (cond ((listp traegt) (setq traegt (finde-objekt traegt objektliste))))
                        (setq nachfolger2 nachfolger)))
    (let ((darueber (get traegt 'traegt-direkt))
          (knoten1 nachfolger2)
          (nachfolger1 (gensym)))
         (anfuegen nachfolger1 knoten1)
         (putprop nachfolger1 (list 'finde-raum objekt traegt) 'situation)
         (cond ((or (equal traegt 'tisch) (null darueber))
                    '(raum ueber ,traegt fuer ,objekt))
               (T nil))))

(defun hole-objekt-unter (ort &optional nachfolger2)
       (cond ((equal nil nachfolger2) (initialisierung)(setq nachfolger2 nachfolger)))
    (let ((knoten1 nachfolger2)
          (nachfolger1 (gensym)))
         (anfuegen nachfolger1 knoten1)
         (putprop nachfolger1 (list 'hole-objekt-unter ort) 'situation)
         (caddr ort)))

(defun oben-auf (objekt &optional nachfolger2)
       (cond ((equal nil nachfolger2) (initialisierung)(setq nachfolger2 nachfolger)))
    (let ((knoten1 nachfolger2)
          (nachfolger1 (gensym)))
         (anfuegen nachfolger1 knoten1)
         (putprop nachfolger1 (list 'oben-auf objekt) 'situation)
         (list 'oben-auf objekt)))

(defun neu-oben-auf (objekt ort &optional nachfolger2)
       (cond ((equal nil nachfolger2) (initialisierung)(setq nachfolger2 nachfolger)))
    (let ((knoten1 nachfolger2)
          (nachfolger1 (gensym)))
         (anfuegen nachfolger1 knoten1)
         (putprop nachfolger1 (list 'neu-oben-auf objekt ort) 'situation)
         (list 'neu-oben-auf objekt ort)))


(defun anfuegen (k e)
    (putprop k e 'knoten)
    (putprop e (append (get e 'nachfolger) (list k)) 'nachfolger))
```

```
(defun sag-wie (situation)


    (wiel (list nachfolger) situation))

(defun wiel (liste situation)
    (cond ((null liste) nil)
              ((equal situation (get (car liste) 'situation))
                  (mapcar #'(lambda (e) (print (get e 'situation)))
                        (get (car liste) 'nachfolger))
               T)
          (T (wiel (append (get (car liste) 'nachfolger) (cdr liste)) situation))))


(defun sag-weshalb (situation)
    (weshalb1 (list nachfolger) situation))

(defun weshalb1 (liste situation)
    (cond ((null liste) nil)
          ((equal situation (get (car liste) 'situation))
                  (print (get (get (car liste) 'knoten) 'situation))
                  T)
          (T (weshalb1 (append (get (car liste) 'nachfolger) (cdr liste)) situation))))

(defun sag-warum (situation)
    (warum1 (list nachfolger) situation))

(defun warum1 (liste situation)
    (cond ((null liste) nil)
          ((equal situation (get (car liste) 'situation))
                (print (get nachfolger 'situation)) T)
          (T (warum1 (append (get (car liste) 'nachfolger) (cdr liste)) situation))))

(defun ausgabe (plan)
      (pp (reverse plan)))
```

Listing 1

Salve Megamaxus

Interrupti te salutant

Interrupts in Megamax-C

In modernen Computern wie dem Atari ST spielen durch Interrupt gesteuerte Funktionen eine wichtige Rolle, da sie die Leistungsfähigkeit des Systems beträchtlich erhöhen können. Der weitverbreitete Megamax-C-Compiler gestattet durch die Verwendung von in-line-Assemblercode eine bequeme Einbindung von Interruptroutinen in C-Programme. Probleme treten leider dann auf, wenn man in dieser Interruptroutine auf globale Variablen dieses C-Programms zugreifen möchte. Doch die Lösung ist einfach und ermöglicht es sogar, die Interruptfunktion komplett in C zu schreiben.

Im Atari ST gibt es zahlreiche Quellen von Interrupts. Serielle und parallele Schnittstelle, verschiedene Timer, die Tastatur oder auch der SHIFTER (der Video-Controller) unterbrechen in regelmäßigen oder unregelmäßigen Abständen den Prozessor, um anzuzeigen, daß sie in irgendeiner Form bedient werden wollen. Ein Drucker an der Parallelschnittstelle kann so z. B. mitteilen, daß er wieder bereit ist, neue Zeichen anzunehmen. Wegen dieser Signale müssen Programme nicht ständig die Peripherie abfragen, um solche Ereignisse nicht zu verpassen. Das spart natürlich Zeit.

Aus Platzgründen ist es leider nicht möglich, an dieser Stelle alle Grundlagen der Interruptprogrammierung zu wiederholen. Hier geht es lediglich um die Realisierung von interruptgesteuerten Programmen in Megamax-C. Die zahlreichen Beispiele können vielleicht aber auch Programmierern in anderen Sprachen Anregungen vermitteln.

Bei Auftreten eines Interrupts unterbricht der Prozessor seine momentane Arbeit, legt den Inhalt des Statusregisters auf dem Stack ab und springt in die Interrupt-Service-Routine, also das Unterprogramm, das zur Bearbeitung dieses Interrupts vorgesehen ist. Dieses Unterprogramm muß dafür sorgen, daß bei Beendigung der Routine die Prozessorregister den gleichen Inhalt haben wie beim Eintritt. Sonst könnte es passieren, daß nach dem Rücksprung das Hauptprogramm nicht mehr korrekt fortgesetzt werden kann (z. B. wenn ein wichtiges Zwischenergebnis in einem Register überschrieben wurde). Der Rücksprung selbst muß durch den Assemblerbefehl 'rte' erfolgen, der dafür sorgt, daß auch das Statusregister wieder mit dem Wert geladen wird, der anfangs auf dem Stack abgelegt wurde. Eine dritte Voraussetzung von Interruptroutinen ist, daß sie keine Annahmen über den Inhalt von Prozessorregistern machen dürfen. Diesem Punkt kommt beim Megamax-Compiler eine besondere Bedeutung zu, die wir gleich sehen werden.

Betrachten wir zuerst die Formulierung einer Interruptroutine in Assembler. Register, die von dieser Funktion benötigt werden, rettet man am besten gleich zu Anfang auf den Stack. Mit dem Befehl'movem.l'(Reg.liste),–(A7)' kann man gleich mehrere – oder auch

alle – Register auf einmal sichern (A7 ist bekanntlich der Stackpointer im MC68000). Der umgekehrte Befehl 'movem.L (A7)+, (Reg.liste)' stellt am Ende der Funktion den alten Inhalt der Register wieder her und ein 'rte' bewirkt den Rücksprung (vgl. Abb. 1):

```
ir _funktion()
{   asm
    {   movem.l D0-D7/A0-A5,-(A7)    ;sichert alle Register auf dem Stack
        ...                          ;hier steht die eigentliche Funktion
        movem.l   (A7)+,D0-D7/A0-A5  ;holt die alten Registerinhalte
        unlk    A6
        rte                          ;beendet die Funktion
    }
}
```

Abb.1

Auffällig ist die merkwürdige Behandlung des Registers A6. Dieses Register wird vom Compiler verwaltet. Schaut man sich die Übersetzung von *ir_funktion()* mit dem Disassembler an, so zeigt sich etwa folgende Sequenz: (vgl. Abb 2):

```
ir_funktion:  link     A6,#0               ;Schafft Platz für lokale Variablen
              movem.l  D0-D7/A0-A5,-(A7)
              ...
              movem.l  (A7)+,D0-D7/A0-A5
              unlk     A6                  ;holt alten Wert von A6
              rte                          ;vom Stack
              unlk     A6                  ;hierhin kommt man nie
              rts                          ;wegen 'rte'
```

Abb.2

Die Verwendung von A6 ist völlig transparent, d. h. das Register hat nach der Funktion den gleichen Wert wie vorher. Daher können in Interruptroutinen lokale Variablen problemlos benutzt werden (näheres steht im Megamax-Handbuch, S. 2-3/2-4, (vgl. Abb. 3):

```
ir_funktion()
{   int lokale_var;
    asm
    {   movem.l D0-D7/A0-A5,-(A7)    ;sichert alle Register auf dem Stack
        move.w  #1,lokale_var(A6)    ;gibt lokale_var den Wert 1
        ...                          ;hier steht die restliche Funktion
        movem.l (A7)+,D0-D7/A0-A5    ;holt die alten Registerinhalte
        unlk    A6
        rte                          ;beendet die Funktion
    }
}
```

Abb.3

Etwas anders ist die Situation, wenn auf globale Variablen zugegriffen werden soll, damit etwa Resultate der Interruptroutine von C-Funktionen weiterverarbeitet werden können. Ein Beispiel biete das Listing 'MSTIMER.C', auf das wir noch zu sprechen kommen.

```
long get_a4()
{  asm
   { move.l A4,D0 }   ;in D0 wird der Funktionswert an das
}                     ;aufrufende Programm zurückgegeben
```

Abb.4

Globale und Stringvariablen werden in Megamax-C ähnlich wie lokale Variablen angesprochen. Nur zeigt hier das Adressregister A4 auf den Speicherbereich, in dem diese Variablen abgelegt sind. A4 hat während der gesamten Programmdauer den gleichen Wert – anders als Register A6 für lokale Variablen, das seinen Wert erst bei Eintritt in eine Funktion erhält. A4 wird beim Programmstart initialisiert. Und hier liegt das Problem für Interruptroutinen: Man kann ja nicht vorhersagen, wann ein Interrupt auftritt und wann demnach die zugehörige Routine aufgerufen wird. Das kann auch während der Abarbeitung eines Programmstücks geschehen, welches das Register A4 temporär mit einem anderen Wert belegt hat (z. B. eine andere Interruptroutine, ein externes Programm oder eine Betriebssystemfunktion). Kurzum, wir können nicht sicher sein, daß A4 in unserer Interruptroutine tatsächlich auf die globalen Variablen zeigt und brauchen daher eine andere Möglichkeit, Daten zwischen C und solchen Routinen auszutauschen.

```
#define IR_ENTRY asm { movem.l D0-D7/A0-A5,-(A7)\
                       movea.l _a4,A4\
                       bra.s   _ok\
                  _a4: dc.l    0L\
                  _ok: }

#define RTE      asm { movem.l (A7)+,D0-D7/A0-A5\
                       unlk    A6\
                       rte }
```

Abb.5

Eine der Möglichkeiten soll im Folgenden kochrezeptartig dargestellt werden. Zur Verdeutlichung dient das Modul TIMER.C, das eine Stopuhr mit einer Auflösung von 1 ms realisiert. In den Zeilen 23..43 können Sie das Vorgehen mitverfolgen:

1. Im Assemblercode wird Speicher zur Aufnahme der Daten reserviert, hier für ein Langwort (= Adresse) => *dc.l 0L*. Mehrere solcher Definitionen, auch für Wort- oder Bytegrößen, sind natürlich möglich.
2. Diese Datenbereiche werden mit Labeln versehen => *_msadr:*.
3. Die Label müssen global deklariert werden => *static void _msadr()* oder *extern void _msadr()*. Wichtig ist die Deklaration als Funktion, da nur diese von Megamax akzeptiert wird. Der Typ der Funktion ist dagegen beliebig. Sie können statt void also auch long oder etwas anderes schreiben.

4. Irgendwo sollte eine korrespondierende C-Variable definiert sein. Bei uns ist das Zeile 25: *static long millisec;*. Static dient nur dazu, diese Variable vor anderen Modulen zu verbergen und kann auch entfallen.

5. Funktionsnamen sind Adressen von Funktionen, also Zeiger. Die Adressen von Variablen sind ebenfalls Zeiger. Zeiger kann man einander zuweisen!

6. Mittels cast-Operator werden die Typen der beiden Zeiger angeglichen (Megamax ist da recht pingelig) => *★(long ★★)__msadr = &millisec*. Man kann auch schreiben: *★(long ★)__msadr = (long) &millisec*.

Das alles hört sich komplizierter an, als es ist. Vergleichen Sie das Listing.

In diesem Beispiel haben wir nur eine Variable der Interruptroutine behandelt. Natürlich könnte man mit dieser Methode auch mehrere Variablen übergeben, aber das wird umständlich. Genausogut kann man auch den Wert des Registers A4 eintragen, über das ja alle globalen Variablen angesprochen werden können. Diesen Wert erfährt man mit folgender kleinen Funktion (Abb. 4):

Endlich haben wir alles beisammen, um Interruptroutinen gänzlich in C zu formulieren. Gänzlich? Nun ja, ein bißchen Assembler bleibt schon übrig. Sie erinnern sich, daß wir bei Funktionseintritt die Inhalte aller benötigten Register retten müssen, um sie am Ende wieder zurückschreiben zu können (das unterbrochene Programm darf ja nicht merken, was in der Zwischenzeit geschehen ist), und daß wir die Funktion ausdrücklich mit 'rte' verlassen müssen, weil das 'rts', das Megamax einsetzt, kein ordnungsgemäßer Abschluß einer Interruptroutine ist. Kurzum, wir brauchen das gleiche Gerüst, das wir oben im Beispiel der *ir__funktion()* gezeigt haben. Man faßt es am besten in zwei Makros, die dann immer zu Beginn und am Ende einer Interruptfunktion aufgerufen werden (wer mag, schreibt sich diese in eine #include-Datei und kann die Details der Implementierung danach beruhigt wieder vergessen). Die Makros finden Sie in Abb. 5

Bitte beachten Sie jeweils den Backslash („\") am Zeilenende, der dem

```
23: static long millisec;
    static long ms_save;
    static void _a4();
    #define IR_ENTRY ...                   /* wie oben beschrieben    */
    #define RTE                            /* auch wie oben           */
    static long get_a4()
    | ... |                                /* ebenfalls schon bekannt */

    static void _timeavec()
    |   char *mpfisra = (char *)=0xfffa0f;

        IR_ENTRY
        millisec++;
        *(mpfisra) &= 0xdf;                /* = bclr #5,MFPISRA */
        RTE
    |
    void ti_on()
    |   ti_stop();
        ti_clr();
        *(long *)_a4 = get_a4();           /* wert von A4 in _timavec() eintragen
                                              für Zugriffg auf millisec       */
        xbtimer(TIMER_A,TIMACTRL;TIMADAT,_timavec);
    |
```

Abb.6

den 'movem.1'-Anweisungen auf die tatsächlich benötigten Register zu verkürzen (A4 gehört auch dazu!). Auf jeden Fall sollten Sie nicht vergessen, den Inhalt von A4 nach dem beschriebenen Vorgehen an der Stelle *__a4:* einzutragen.

Mit diesen Hilfen könnten wir die Zeilen 23..43 im Beispielprogramm nun auch so schreiben, wie es Abb. 6 zeigt.

Der Zuwachs an Übersichtlichkeit schaltet mögliche Fehlerquellen aus, und von ihnen gibt es ohnehin genug.

Abschließend noch einige kurze Worte zu dem Beispielprogramm MSTIMER.C:

Kernstück ist eine Routine, die von Interrupts des (im ST sonst nicht benutzten) Timers A aufgerufen wird. Dieser wird so programmiert, daß er etwa jede Millisekunde eine Unterbrechung auslöst (genau läßt sich das leider nicht erreichen). Die Funktion wird also sehr oft abgearbeitet und sollte daher möglichst schnell sein, um die Systemleistung nicht zu arg herabzusetzen.

```
 1:  /*************************************************
 2:  *                  MSTIMER.H                    *
 3:  *       Headerfile zum Millisekunden-Timer      *
 4:  *          Megamax C, V1.1 auf Atari ST         *
 5:  *************************************************
 6:  *   H. Huptasch, Deisterstr. 15, 3250 Hameln    *
 7:  *************************************************/
 8:
 9:  extern long ti_get();
10:  extern long ti_cget();
11:  extern long ti_msget();
12:  extern long ti_mscget();
13:  extern long ti_conv();
```

Compiler mitteilt, daß die Makrodefinition in der nächsten Zeile fortgesetzt wird. Register A6 wird – wie erwähnt – beim Eintritt in die Funktion vom Compiler verwaltet, beim Verlassen müssen wir selbst den alten Wert restaurieren. Um die Makros universell zu halten, werden einfach alle Register auf dem Stack zwischengelagert. Nur bei sehr zeitkritischen Interruptroutinen lohnt es sich, die Registerliste in Deshalb wird als Adresse der Interruptroutine auch nicht die der C-Funktion angegeben (die darum *dummy()* heißt), sondern das Label *_timavec:* innerhalb der Funktion. Man spart damit die Reservierung von Speicher für lokale Variablen, die hier ohnehin nicht benötigt werden (vgl. Megamax-Handbuch S. 2-3). Somit muß auch am Ende der Funktion nicht erst der Inhalt von A6 restauriert werden. Auch das Eintragen der Adresse des ms-Zählers in den Programmcode ist nicht lehrbuchreif, aber schnell. Vorteil: Wenn diese Interruptroutine mitläuft, reduziert sich die Systemgeschwindigkeit trotz des häufigen Aufrufs nur um knapp ein Prozent.

Die weiteren Funktionen sind sehr einfach. *Ti_on()* schreibt die Adresse des Zählers in die Funktion, initialisiert den Timer A auf eine Frequenz von ca. 1 kHz, trägt das Label *_timavec:* als Adresse der Interruptroutine ein und startet den Timer dann – alles mit dem Aufruf der Betriebssystemfunktion *Xbtimer()*. *Ti_cget()* bzw. *ti_mscget()* geben die Zeit seit dem (Re-) Start des Timer wieder (das c steht für cumuliert), *ti_get()* bzw. *ti_msget()* ergeben die Zeit seit der letzten Abfrage, damit können also Zwischenzeiten gemessen werden. Die Funktionen mit *ms* im Namen verrechnen die ermittelten Zeiten noch mit einem Korrekturfaktor, so daß die Zeitbasis tatsächlich einer Millisekunde entspricht. Mit *ti_restart()* wird der ms-Zähler auf Null gesetzt, und *ti_off()* sollte immer aufgerufen werden, wenn der Timer nicht immer benötigt wird, unbedingt aber am Programmende. Diese Funktion schaltet nämlich die Interrupts des Timers A wieder ab. Stellen Sie sich einmal vor, was passiert, wenn nach Programmende so ein Interrupt auftritt, und die zugehörige Interruptroutine steht nicht mehr im Speicher: Natürlich gibt das eine „Bombenstimmung". In der #include-Datei MSTIMER.H werden die Funktionen als extern deklariert. Das ist nötig, da sie nicht vom Typ 'int' sind.

Viel Spaß und Erfolg beim Experimentieren also, und wenn's mal schief geht: Abstürze gehören einfach dazu!

H. Huptach

```
 1:  /*************************************************
 2:  *                  MSTIMER.C                    *
 3:  *      Timer mit 1 Millisekunde Auflösung       *
 4:  *         Megamax C, V1.1 auf Atari ST          *
 5:  *************************************************
 6:  *   H. Huptasch, Deisterstr. 15, 3250 Hameln    *
 7:  *************************************************/
 8:
 9:  #include <osbind.h>
10:  #define void int
11:
12:  #define MFPISRA 0xfffa0f
13:              /* MPF Interrupt-In-Service Reg. A */
14:  #define TIMER_A 0
15:  #define TIMADAT 49          /* Zähler für Timer A */
16:  #define TIMACTRL 4      /* Timer A, Vorteiler 50 */
17:  #define MFP_TIMERA 13
18:                /* MFP interrupt level, timer A */
19:
20:  #define ti_stop() Xbtimer(TIMER_A,0,0,-1L)
21:  #define ti_clr() (millisec = ms_save = 0L)
22:
23:  static long millisec;
24:  static long ms_save;
25:  static void _timavec();
26:  static void _msadr();                /* Trick 17! */
27:
28:
29:  static void dummy()
30:  {  asm
31:     {  _timavec:  dc.w  0x52b9  ;addq.l #1,$xxxxxx
32:        _msadr:    dc.l  0L      ;Adr. ms-Zähler
33:                   bclr  #5,MFPISRA ;IRQ löschen
34:                   rte
35:     }
36:  }
37:
38:  void ti_on()       /* Initialisierung und Start */
39:  {  ti_stop();                /* Vorsichtshalber... */
40:     ti_clr();
41:     *(long **)_msadr = &millisec;   /* Trick 17! */
42:      /* Adr. des Zählers in IR-Routine eintragen */
43:     Xbtimer(TIMER_A,TIMACTRL,TIMADAT,_timavec);
44:  }
45:
46:  void ti_off()  /* Stoppen und Interrupt disablen*/
47:  {  ti_stop();
48:     Jdisint(MFP_TIMERA);
49:  }
50:
51:  void ti_restart()      /* Zähler auf Null setzen */
52:  {  ti_clr();  }
53:
54:  long ti_get()        /* Zeit seit letzter Abfrage */
55:  {  register long t;
56:
57:     t = millisec - ms_save;
58:     ms_save = millisec;
59:     return t;
60:  }
61:
62:  long ti_cget()  /* Zeit seit letztem (Re-)Start */
63:  {  return millisec;  }
64:
65:  #define ADJUST (49.0/49.152) /* Korrekturfaktor */
66:
67:  long ti_msget()    /* ti_get() in Millisekunden */
68:  {  return (long)((ti_get() * ADJUST)); }
69:
70:  long ti_mscget()  /* ti_cget() in Millisekunden */
71:  {  return (long)((ti_cget() * ADJUST));  }
72:
73:  long ti_conv(t)   /* Timerwerte in ms umrechnen */
74:     long t;
75:  {  return (long)((t * ADJUST));  }
```

Der Maus auf die Sprünge geholfen

Eine Maus hat es schwer

Manch einem ist der Auslauf der Maus zu groß, eben weil der Schreibtisch zu klein ist. Andere hätten gerne einen größeren Bereich um genauere Zeichnungen anzufertigen. Das folgende Listing zeigt, wie man das in Modula ändern kann.

Allgemeines

Nach einer Verdoppelung der Mausgeschwindigkeit benötigt man nur ein Viertel der sonst üblichen Tischfläche, um die Maus über den gesamten Bildschirm zu bewegen, so daß man alle Programme aus dem Handgelenk steuern kann, ohne den Handballen vom Tisch zu heben.

Nachdem Sie einige Zeit mit diesem Accessory gearbeitet haben, werden Sie sicher gerne 1.5 KByte Ihres Hauptspeichers spendieren, um wesentlich zügiger und komfortabler mit Ihren GEM-Programmen zu arbeiten.

Bedienung

Nach dem Booten des Systems bewegt sich der Mauscursor am Bildschirm doppelt so schnell wie üblich. Außerdem können Sie im Menü DESK unter dem Menüpunkt mouse speed jederzeit folgende Parameter ändern:

SPEED : 1 2 3 (normale, doppelte, dreifache Geschwindigkeit)

```
 1:   MODULE SpeedMouse;
 2:   (*
 3:      Florian Matthes   12.5.1987
 4:      Beschleunigung der Maus als Accessory
 5:      Beim Linken GEMACCX.LNK statt GEMX.LNK verwenden!
 6:
 7:   *)
 8:
 9:   (*$T-,$S- (Stacktest etc. ausschalten) *)
10:
11:   IMPORT XBIOS;
12:   FROM SYSTEM          IMPORT ADR, ADDRESS, REGISTER, SETREG, CODE, BYTE;
13:   FROM AESApplications IMPORT ApplInitialise;
14:   FROM GEMAESbase      IMPORT AccessoryOpen, MesageEvent;
15:   FROM AESForms        IMPORT FormAlert;
16:   FROM AESMenus        IMPORT MenuRegister;
17:   FROM AESEvents       IMPORT EventMessage;
18:
19:   CONST
20:     Title = "  Mouse Speed";
21:
22:   TYPE
23:     MousePackagePtr = POINTER TO
24:                         RECORD
25:                           ButtonState,DeltaX,DeltaY : BYTE;
26:                         END;
27:   VAR
28:     applID    : INTEGER;                   (* GEM application ID  *)
29:     menuID    : INTEGER;                   (* menu ID             *)
30:     Msg       : ARRAY [0..16] OF INTEGER;  (* message buffer      *)
31:     x         : CARDINAL;
32:
33:     (* globale Variablen für den neuen Interrupt-Handler:          *)
34:     Trigger, Speed: CARDINAL;
35:     KBVec         : XBIOS.KBVectorPtr;
36:     PackagePtr    : MousePackagePtr;
37:     oldvec        : PROC;
38:     Count         : CARDINAL;
39:
40:   (*$P-  Verhindere Erzeugung des Modula Prozedur-Eintritts und Ende-Codes,
41:          da die folgende Prozedur direkt aus Maschinensprache aufgerufen
42:          werden soll
43:   *)
44:
45:   PROCEDURE MouseMovement;
46:   (* Diese Prozedur wird von XBIOS aufgerufen, falls dieses ein 'mouse package'
47:      von dem 'intelligent keyboard processor' erhält, das anzeigt, daß die
48:      Maus bewegt wurde. Das CPU-Register A0.L zeigt auf ein package im folgen-
49:      den Format:
50:           0:   Zustand der Maustasten (Bit 0 und 1)
51:           1:   X-Offset relativ zur letzten Position (-128..127)
52:           2:   Y-Offset relativ zur letzten Position (-128..127)
53:   *)
54:   BEGIN
```

TRIGGER: 0 1 2 (kein Trigger, Trigger = 1 oder 2 Pixel)

Der Trigger ermöglicht es, die Beschleunigung des Mauscursors bei sehr kleinen Bewegungen automatisch zu unterdrücken, so daß selbst bei einer dreifachen Geschwindigkeit ein pixelgenaues Arbeiten am Bildschirm möglich ist.

Funktionsweise

Im XBIOS existiert eine Routine, die immer dann aufgerufen wird, wenn von dem Tastaturprozessor des ATARI ST gemeldet wird, daß der Benutzer die Maus bewegt hat. Dabei befindet sich in dem Prozessorregister A0 ein Zeiger auf einem Datenblock im Speicher, der neben dem Zustand der Maustasten (gedrückt / nicht gedrückt) die **relative** Bewegung der Maus in X und Y-Richtung seit der letzten Meldung enthält.

Um nun die Mausgeschwindigkeit zu verdoppeln, genügt es, diese Routine für jede Bewegung der Maus zweimal aufzurufen. Durch einen Vergleich der relativen Bewegung mit der Variablen Trigger wird ein mehrfacher Aufruf verhindert, falls die Maus nur ein kleines Stückchen bewegt wurde.

Interessant für Insider ist vielleicht noch die Realisierung einer Assemblerroutine (MouseMovement) in Modula-2: Zunächst wird mit der Compiler-Direktive $P– die Erzeugung des üblichen Prozedurprologs und -epilogs verhindert. Am Beginn und Ende der Prozedur werden die verwendeten Register mit Assemblerbefehlen auf dem Stack gerettet, von wo die Register gezielt (mit stackrelativer Adressierung) geladen werden können. Außerdem zeigt die Variable oldvec, wie der (indirekte) Aufruf einer parameterlosen Assemblerroutine von Modula-2 aus möglich ist.

Die Standardvorgaben für Geschwindigkeit und Trigger (2 und 0) können Sie in der 3. Zeile des Hauptprogrammes modifizieren.

Übrigens läßt sich der erzeugte Maschinencode mit dem Hilfsprogramm DECLNK.PRG disassemblieren, um z.B. die verwendeten Register in einer Prozedur zu bestimmen.

```
 55:     (* rette benötigte Register auf den Stack: *)
 56:     CODE(048E7H, 08C88H);        (* MOVEM.L D0/D4/D5/A0/A4, -(A7)           *)
 57:     CODE(0202FH, 0000CH);        (* MOVE.L  12(A7), D0   (= A0 )            *)
 58:
 59:     PackagePtr:= REGISTER(0);  (* hole Inhalt von D0, zerstört auch D4   *)
 60:     IF    (CARDINAL(PackagePtr^.DeltaX)>Trigger)
 61:       AND (CARDINAL(PackagePtr^.DeltaX)<256-Trigger)
 62:     OR
 63:           (CARDINAL(PackagePtr^.DeltaY)>Trigger)
 64:       AND (CARDINAL(PackagePtr^.DeltaY)<256-Trigger)  THEN
 65:       Count:= Speed;              (* rufe die alte Routinen Speed-mal auf: *)
 66:       REPEAT
 67:         oldvec;                   (* zerstört das Register A4              *)
 68:         DEC(Count);
 69:       UNTIL Count=0;
 70:     ELSE
 71:       oldvec;
 72:     END;
 73:
 74:     (* restauriere alle Register vom Stack:                                *)
 75:     CODE(04CDFH, 01131H);        (* MOVEM.L (A7)+, D0/D4/D5/A0/A4          *)
 76:     CODE(04E75H);                (* RTS                                    *)
 77:   END MouseMovement;
 78:
 79:   (*$P+ Rückkehr zum 'normalen' Modus der Codeerzeugung                    *)
 80:
 81:   BEGIN
 82:     applID := ApplInitialise();
 83:     menuID := MenuRegister(applID,Title);
 84:
 85:     (* Voreinstellung: Kein Trigger, doppelte Geschwindigkeit:             *)
 86:     Trigger:= 0; Speed:= 2;
 87:     (* hole Zeiger auf Tabelle der XBIOS-Routinen                          *)
 88:     KBVec:= XBIOS.KeyboardVectors();
 89:     WITH KBVec^ DO
 90:       (* merke ursprüngliche Adresse des Interrupthandlers:                *)
 91:       oldvec:= PROC(mousevec);
 92:       (* leite Interrupts zur Routine MouseMovement um:                    *)
 93:       mousevec:= ADDRESS(MouseMovement);
 94:     END;
 95:
 96:     (* Erlaube Änderung der Parameter durch den Benutzer:                  *)
 97:     LOOP
 98:       EventMessage(ADR(Msg));      (* warte, bis Benutzer Acessory wählt *)
 99:       IF Msg[0] = AccessoryOpen THEN
100:         x := FormAlert(2,
101:           "[2][Which parameter do you want to modify?][ Trigger | Speed ]") ;
102:         IF x = 1 THEN
103:           Trigger:= FormAlert(Trigger+1,
104:             "[2][Trigger for speedup (in pixel)?][ 0 | 1 | 2 ]") - 1;
105:         ELSE
106:           Speed := FormAlert(Speed,"[2][Mouse speed factor?][ 1 | 2 | 3 ]");
107:         END;
108:       END;
109:     END;
110:   END SpeedMouse.
```

Listing 1: Mausspeed in Modula-2

Übersetzen des Quelltextes

Nach dem Abtippen übersetzen Sie das Programm wie üblich. Zum Linken wählen Sie neben der Option OPT (für Optimieren) die Option QUERY, so daß Sie bei der Frage des Linkers nach dem File GEMX.LNK das File GEMACCX.LNK eingeben können. Dieses File enthält ein sehr kompaktes Modula-Laufzeitsystem für Accesories. Das entstehende Programm (z. B. SPEEDMOU.PRG) speichern Sie auf der Bootdiskette unter dem Namen SPEED.ACC.

Florian Matthes

Relax

KNIGHT ORC

Es ist nicht leicht ein ORC zu werden, allerdings ist es die einzige Möglichkeit, dieses Spiel zu bestehen. ORCs sind feige, hinterhältig, faul und gefräßig. Ausserdem halten sie nicht viel von der Spezies Mensch - um ehrlich zu sein, sie hassen sie zuweilen. Gerade jetzt ist wieder solch ein Moment: einige der ORKs haben sich auf den Weg gemacht, einen Menschen den Garaus zu machen. Doch der Überfall endet in einem fürchterlichen Besäufnis. Als die ORCs ihren Rausch ausschlafen, werden sie von Sir Cecils Privatarmee überfallen und zu einem "Kampf der Besten" herausgefordert. Doch auch der heftigste Rausch hat einmal ein Ende und das ist am folgenden Morgen der Fall. Den ORCs wird schmerzhaft bewußt, auf was sie sich da eingelassen haben. Feige wie sie nun mal sind, beschließen sie, zu flüchten. Nur einer bleibt zurück: Grindleguts. Dieser liegt noch immer unter dem Tisch und ist noch nicht bei Sinnen. Als Sir Cecil eintrifft, ist er immer noch da, denn seine Freunde haben ihn am Tisch festgebunden. Nun kommt ihr Part, denn Sie müssen von nun an die Rolle des armen Grindleguts übernehmen, der versuchen muß, die nun beginnenden Geschehnisse zu überstehen. Doch einfach ist es nicht. Die Szene beginnt mit einem Reiterturnier, bei dem Sie aus Sicherheitsgründen an ihr Pferd gebunden sind.

Dem Programmiererteam LEVEL 9 ist es mit KNIGHT ORC gelungen, ein skurilles Grafikadventure zu entwerfen, das spannende und erheiternde Unterhaltung bietet. Ein besonders Lob verdient auch der Parser, der nicht nur ganze Sätze und Anweisungsfolgen erkennt, sondern auch Befehle wie GO TO... und FIND... selbständig ausführt. Außerdem kann in besonderem Maß mit anderen Figuren kommuniziert werden. Sie können, wenn man sie dazu gebracht hat, bestimmte Anweisungen ausführen, wobei auch ein Zusammenwirken mehrerer Akteure möglich und zur Lösung notwendig ist. Bei der Grafik handelt es sich um Umsetzungen von Bleistiftzeichnungen, so daß das Bild verschwommen erscheint. Dies könnte daran liegen, daß sie sowohl auf dem Monochrom- als auch auf einem Farbmonitor darstellbar ist. Den stimmungsvollen Einstieg in die Welt der ORCs vermittelt die knapp 50-seitige Novelle, die als deutsche Übersetzung vorliegt.

(mn)

KNIGHT ORC

Kategorie: Grafik-Adventure
Besonderheiten: guter Parser, scurille Story, dt. Anleitung, Novelle
Spieler: 1
Monitor: color/mono
Steuerung: Tastatur
Hersteller: Rainbird
Vertrieb: Ariola Soft, Tel 05241/803871
Preis: ca. 59,95 DM
Wertung:

DIGGER

Digger ist ein Strategiespiel, das viel Geduld erfordert. Man ist ein kleines Männchen in einem Labyrinth, welches aus Mauern, Erde, Steinen und Goldstücken besteht. Das Männchen soll nun alle vorhandenen Goldstücke einsammeln.

Zuerst macht man sich einen genauen Plan, da man sonst den Überblick verliert. Schon der erste Level ist so schwierig, daß man dafür einige Zeit aufwenden muß. Die Erde, für die man auch Punkte bekommt, muß abgetragen werden, um für die herunterfallenden Steine Platz zu machen. Aber Vorsicht ist geboten, denn schnell ist man eingeklemmt und muß neu beginnen. Das Männchen wird nicht mit dem Joystick gesteuert, sondern mit der Tastatur. Wer Zeit hat und sich nicht davon abschrekken läßt, öfters von vorne zu beginnen, der wird Gefallen an "Digger" finden.

(AS)

DIGGER

Kategorie: Strategiespiel
Besonderheiten:
Spieler: 1
Monitor: color
Steuerung: Tastatur
Hersteller: Stone-Castle
Vertrieb: Data Beutner KG
Preis: ca.
Wertung:

BASKETBALL

Basketball, auch Korbball genannt, ist eigentlich ein Sport der Langen, doch auf dem Computer hat das keine Bedeutung. Basketball auf dem ST kann man zu zweit oder alleine spielen. Man sieht das Spielfeld immer nur von einer Hälfte aus. Zwei Angreifer stehen zwei Abwehrenden gegenüber. Die hellere Figur steuert man mit dem Joystick, die andere wird vom Computer gesteuert. Für einen Angriff hat man nur begrenzte Zeit zur Verfügung. Man sollte nicht zuviel tribbeln und wenn man auf den Gegener aufläuft gibt es ein "Stürmerfaul". Auch der Mitspieler, der vom Computer gesteuert wird, versucht den Ball in den Korb zu spielen. Hält dieser den Ball und der Spieler mit seiner Figur steht unter dem Korb, versucht er, ihn anzuspielen. Deshalb ist es sinnvoll, den zweiten Mann nicht auszuschliessen, sondern ins Spiel zu integrieren. Selbst in der Anleitung steht, daß Basketball ein Spiel ist, in welchem man nur mit 'Teamwork' ausharren kann.
Eine Besonderheit ist auch das Spielen in der Basketball-Liga. Zuerst wird ausgewählt in welcher Liga man spielen möchte: Nord, Süd, Ost oder West, je nach Schwierigkeitsgrad. Gespielt wird dann mit fünf anderen Mannschaften in einer Fünf-Spiele-Saison gegeneinander. Schließt man die Runde als Sieger ab, steigt man auf in die Play-Off-Runde.
Basketball ist ein gutes Sportspiel, bei dem auch die Kleinen beweisen können, daß sie gute Werfer sind.

(AS)

BASKETTBALL

Kategorie: Sportspiel
Besonderheiten: WettKampfspiel

Spieler: 1 ... 11
Monitor: color
Steuerung:
Hersteller: ACTIVISION
Vertrieb: profisoft GmbH

Preis: ca. 69.95 DM

Wertung:

SUB BATTLE

SUB BATTLE ist ein U-Boot Simulator, der sich auf die Ereignisse im II. Weltkrieg bezieht. Zuerst wird ein Land gewählt (entweder Deutschland, oder Amerika). Danach bekommt man eine Mission zugewiesen. Wurde Deutschland gewählt, kämpft man gegen Amerika und England, meist im Nordatlantik; wurde Amerika gewählt, so kämpft man gegen Japan im Pazifik. Die Aufgaben sind die, die gegnerischen Schiffe zu versenken. Ab und an sind in Seenot geratene Menschen zu erretten. Es müssen viele Instrumente kontrolliert werden, der Bildschirm ist hierfür in mehrere Fenster unterteilt. Gesteuert wird mit Joystick und Tastatur. Es gibt eine Fülle von Funktionen; deshalb liegt dem Spiele eine Kurzanleitung bei, auf dem die Tastaturbelegungen und alle anderen Funktionen aufgeführt sind. Auf einer Übersichtskarte kann jederzeit die Position der feindlichen Schiffe, wie auch der eigenen verfolgt werden. Hat man sich einem feindlichen Schiff genähert, muß man auf Seerohrtiefe gehen und das Schiff angreifen. Man kann jedoch nicht nur von feindlichen Schiffen, sondern auch von feindlichen Flugzeugen angegriffen werden. Da gibt es nur noch die Möglichkeit zu tauchen. Schlimm wird es, wenn Wasserbomben geworfen werden, dann gelingt es nur noch mit Geschick den Feinden zu entkommen. Sehr nützlich ist es auch, wenn man sich gelegentlich über den Zustand des Bootes informiert, auf einen Tastendruck erscheint die Angabe der Menge, des vorhandenen Treibstoffes, des Sauerstoffes, und des Stromvorrates.
SUB BATTLE ist eine gelungene U-Boot Simulation mit guter Bedienung. Jedoch handelt es sich hier um ein Spiel mit konkreter Kriegshandlung, weshalb wir das Spiel abgewertet haben.

(AS)

SUB BATTLE

Kategorie: U-Boot Simulation
Besonderheiten: gute Grafik und Bedienung
aber konkrete Kriegshandlung

Spieler: 1
Monitor: color
Steuerung:
Hersteller: EPYX
Vertrieb: profisoft GmbH

Preis: ca. 79 DM

Wertung:

Der ROADRUNNER hat es immer eilig...

Wer kennt ihn nicht, den lustigen Roadrunner, der pfeilschnell durch die Wüste rast. Sein einzigster Feind ist Wile E. Coyote., der mit allen Mitteln versucht, unseren Freund den Roadrunner zu fangen. Kein noch so gemeiner Hinterhalt und jedes Hilfsmittel sind ihm recht, um sein Ziel zu erreichen. Stürzen Sie sich in dieses rasante Spiel und helfen Sie dem 'Raser', seinem Verfolger zu entkommen. Dabei müssen möglichst alle Körnerhaufen aufgegessen werden, denn das bringt Punkte. Der Kojote nimmt schon im zweiten Level ein Jet-Pack zu Hilfe, denn zu Fuß ist er nicht schnell genug. Sein Repertoir umfaßt aber auch noch den Pogo-Stick und ein Jet-Skateboard, mit denen er gefährlich durch die Gegend braust. Es gibt aber noch zusätzliche Hindernisse, wie z. B. LKWs, Felsbrokken, Gletscherspalten und Minen, denen Beide ausweichen müssen. Man kann deshalb auch versuchen, den Kojoten in diese Hindernisse hineinzulocken.

Aus der Umsetzung dieses bekannten (Fernseh)Comics ist ein lustiges Spiel geworden. Resümee: recht kurzweilig, aber nicht sehr aufregend.

(mn)

Road Runner

Kategorie: Actionspiel
Besonderheiten:

Spieler: 1 ... 11
Monitor: color
Steuerung:
Hersteller: US Gold
Vertrieb: Rushware
Tel. 02101/6070
Preis: ca. 69.95 DM

Wertung:

BAD CAT

Los Angeles 1984: Die Stadt rüstet sich für die Olympischen Sommerspiele. Große Leute halten ebensogroße Reden. Am Rande dieses Happenings finden sich die streunenden Großstadtkatzen ein, um ihre eigene Olympiade zu veranstalten. Sie trefen sich überall in der Stadt, um die ausgefallensten Wettbewerbe auszutragen. Der Favorit ist BAD CAT. In rasanten Verfolgungsjagden, stets auf der Flucht vor Hunden, muß zunächst der große Park erreicht werden. Weiter geht es dann in der übelsten Spelunke der Stadt. Beim Kegeln trifft BAD CAT auf die dicke Bulldogge aus dem Westteil der Stadt. Wird er von einer Kugel getroffen, muß er ein großes Bier austrinken. In mehr oder minder ramponiertem zustand geht es danach ins Kanalsystem von Los Angeles, Geschicklichkeit und Mut sind hier besonders gefragt. Endlich im Stadion angelangt, ist Akrobatik Trumpf. Graphisch hebt sich BAD CAT angenehm vom Durchschnitt ab, auch die Titelmusik ist sehr ansprechend. Besonders gut haben uns die vielen, ganz unterschiedlichen Spielsituationen gefallen, in die BAD CAT gerät. Zu Ballerspielen setzt BAD CAT einen angenehmen Kontrapunkt - Geschicklichkeit ist Trumpf.

(cpl)

BAD CAT

Kategorie: Geschicklichkeitsspiel
Besonderheiten:

Spieler: 1 ... 1111
Monitor: color
Steuerung:
Hersteller: Rainbow Arts
Vertrieb: RUSHWARE
Bruchweg 128-132
4044 Kaarst 2
Preis: ca. 59.-

Wertung:

PUZZLE:

EIN VIDEODIGITIZER IM SELBSTBAU

Die Überschrift sagt alles. Dieses Projekt soll Sie mit dem Selbstbau eines Video - Digitalisierers beschäftigen. Bei der Angebotsfülle an Video - Digitalisierern haben wir uns entschlossen unseren Easytizer als Projekt in der ST - COMPUTER zu bringen, zumal so etwas im Selbstbau wohl nicht an der Tagesordnung steht. Im ersten Teil wird die Hardware des Easytizers besprochen und erklärt. Im nächsten Heft geht es dann um die softwaremäßige Ansteuerung der Hardware mit entsprechenden Assemblerroutinen zum digitalisieren eines Bildes. Nichtbastler können schon jetzt ein fertig aufgebautes Gerät samt Software oder auch einen Teilsatz bei der Merlin Computer GmbH bestellen.

Bild 1: Digitalisierter Taschenrechner

Bild 2: Obstschale aus ST Computer Nr. 4 digitalisiert

Bild 3: Macroaufnahme des Firmenlogos, daneben eine Büroklammer

Ein Video-Digitizer im Selbstbau soll nicht gleich bedeuten, daß dieser weniger leisten kann als ein Industriegerät. Die folgenden Leistungsmerkmale zeigen, daß der Easytizer mit so manchen Besonderheiten ausgestattet ist, die einen Nachbau durchaus lohnenswert machen.

Technische Daten:

- Ein Videosignal (0.5–1Vss) kann mit einer Auflösung von maximal 800★600 (horizontal/vertikal) Bildpunkten in bis zu 4 Graustufen **parallel** digitalisiert werden.
- Hohe Geschwindigkeit durch paralleles digitalisieren der vier Graustufen und in Assembler geschriebenen Einleseroutinen. Zeitbedarf: entweder 4 Halbbilder (80 ms) in vier Graustufen mit 640x200 Bildpunkten oder 4 Vollbilder (160 ms) in schwarz-weiß mit 640x400 Bildpunkten. Diese Zeiten sind reale Zeiten, d. h. es vergehen z. B. 80 ms bis das vollständige Bild in den Rechner übertragen ist und auch sichtbar wird. (Das bedeutet 12,5 Bilder pro Sekunde.)
- Gute Störsicherheit durch eingebauten Videoverstärker, so daß auch verrauschte Videosignale oder Signale von Videorecordern noch gut verarbeitet werden können.
- Betrieb am ROM-Port des ST mit einem maximalen Strombedarf von nur ca. 150 mA durch die Verwendung programmierbarer Logik und der Möglichkeit, die ganze Karte aus dem Speicher auszublenden. (Mittels J1)
- Anschluß einer 'Schnappschuß-Taste' ermöglicht das Festhalten eines Bildes von der Kamera aus.
- Wahlweise manuelle oder automatische Weißwerteinstellung, so daß nur ein einmaliger Abgleich an den Kameratyp erfolgen muß.

Aufbau eines Videosignals

Zunächst etwas über die deutsche Videonorm. Zur Übertragung von schwarzweiß-Bildern über einen Informationskanal einigte man sich schon vor langer Zeit auf die noch heute gebräuchliche Form, das **BAS**-Signal (**B**ild-, **A**ustast- und **S**ynchron-Signal). Ein Video-Signal enthält zweierlei Information:

1. Synchroninformation
2. Bildinformation

1.) Die Synchroninformation

Die Synchroninformation dient dazu, einen Gleichlauf des zeilenweise abgetasteten Bildes der Aufnahmeeinheit (z. B. Videokamera) und der Wiedergabeeinheit (z. B. Monitor) zu erzwingen. Dies ist nötig, da es in der Praxis nahezu unmöglich ist, in der Wiedergabeeinheit exakt die selbe Frequenz und die Phasenlage zur Aufnahmeeinheit – besonders auf längere Zeit – zu halten. In der Synchroninformation ist die Zeilenfrequenz und die Bildfrequenz enthalten. Die Zeilenfrequenz (15.625 KHz) ist die Frequenz, mit der z. B. der Elektronenstrahl eines Monitors wieder an den linken Rand und ein Stück nach unten springt, um dann

Abbildung 1: Oben wird eine komplette Videozeile dargestellt. Der Freiraum zwischen Bildsignal und Synchronsignal wird vordere bzw. hintere 'Schwarzschulter' genannt. Bei der Rasterwechselimpulsfolge ist zu beachten, daß 618 die letzte und 11 theoretisch die erste (richtige) sichtbare Zeile ist. Bei den öffentlichen Rundfunkanstalten werden diese ersten sichtbaren Zeilen jedoch oft dazu benutzt, zusätzliche Informationen wie Videotext-Daten zu übertragen. In Wirklichkeit ist ein Fernsehgerät meistens so justiert, daß die wahre Bildgröße über den Rand der Bildröhre hinausgeht und daher etwa die ersten und letzten 20 Zeilen sowie ein kleiner Teil der Zeilen links und rechts verlorengeht.

von links nach rechts eine neue Zeile zu zeichnen. Mit der Bildfrequenz (50 Hz) wiederholt sich eine komplette Bildperiode, die aus 312 beziehungsweise 313 Zeilen (später mehr dazu) besteht und der Elektronenstrahl springt an die linke obere Ecke zurück. Zur Bildwechsel-Synchronisation (Halbbildwechsel) bedient man sich einer speziellen *Rasterwechselimpulsfolge* (siehe auch Abbildung 1), die elektronisch jedoch durch simples Integrieren des abgetrennten Synchronsignals ausgewertet werden kann. Die Synchroninformation liegt pegelmäßig deutlich unter der Bildinformation und kann deshalb mit relativ einfachen Mitteln abgetrennt werden.

2.) Die Bildinformation

Eine Zeile enthält im Verlauf von links nach rechts die Helligkeitsinformation des Bildes. Daher auch die Tatsache, daß die horizontale Bildauflösung nur durch die Bandbreite der Übertragungsstrecke bestimmt wird, die Vertikale jedoch von der Zeilenzahl.
Eine Zeile ist in Zeilensynchronimpulse eingeschlossen. Eines wurde Ihnen bis jetzt jedoch vorenthalten: Sie haben sich vielleicht schon gewundert, warum ein Bild einmal 312 und das andere mal 313 Zeilen haben soll. Das rührt daher, daß man damals einen akzeptablen Kompromiß zwischen Bandbreite der Übertragungsstrecke und der Zeilenzahl (Vertikale Auflösung) zu finden suchte, und es zur Entwicklung des *Interlace-Verfahrens* kam.

Das Interlace-Verfahren

steigert die Zeilenzahl des Video-Bildes auf das Doppelte ohne jedoch die nötige Übertragungsbandbreite zu erhöhen. Das Prinzip beruht einfach darauf, daß man in einem Halbbild nur ungerade Zeilen überträgt und im nächsten Halbbild um einen halben Zeilenabstand versetzt nur gerade Zeilen. Daraus resultiert, daß es jetzt zwei Halbbilder mit einmal 313 und einmal 312 Zeilen gibt. Der Rasterwechselimpulsfolge wird dabei die Aufgabe übertragen, ein Halbbild dem anderen gegenüber jeweils um einen halben Zeilenabstand zu versetzen. Das brachte natürlich Nachteile, denn eine einzige Zeile wird jetzt nur noch mit 25 Hz 'aufgefrischt', die benachbarte Zeile jedoch gleich darauf und damit das ganze Bild wie gehabt mit 50 Hz. Das damit erkaufte Flimmern benachbarter Zeilen zueinander konnte bei normalen 'Fernsehbildern' und aufgrund der zeitlichen Integrierwirkung des menschlichen Auges vernachlässigt werden. Abbildung 1 soll diese Zusammenhänge noch etwas anschaulicher darstellen.

Die Farbinformation im Videosignal

Ein FBAS-Signal (Farb-, Bild-, Austast- und Synchron-Signal) unterscheidet sich von einem BAS-Signal lediglich durch eine in quadraturmodulation aufmodulierte Farbinformation mit einer Trägerfrequenz von 4,43 MHz und dem sogenannten Farbburst, der direkt auf das Synchron-Signal folgt. Bei diesem Burst handelt es sich um etwa zehn Perioden der Farbträgerfrequenz, die zur Synchronisation des Quadraturdemulators benötigt werden. Da der EASYTIZER ein schwarzweiß-Digitizer ist und diese Farbinformation nicht auswerten kann, soll hierauf nicht weiter eingegangen werden. Jedoch wollen wir auf einen wichtigen Punkt hinweisen: Diese dem Bildsignal überlagerte Farbträgerfrequenz macht sich unter Umständen als störendes Muster auf dem digitalisierten Bild (Moiré) bemerkbar. Abhilfe schafft entweder ein Farbträgersperrfilter oder die Verwendung einer reinen schwarzweiß Kamera.

Video und Computer

Um Videosignale elektronisch zu verarbeiten, muß man schon etwas mehr Aufwand treiben als z. B. bei Audio-Signalen. Die ganze Elektronik des EASYTIZER's konnte jedoch auf einer nur 100★100 mm doppelseitig kaschierten Platine untergebracht werden, was dem Einsatz von relativ hochintegrierten Logik-Bausteinen und einem *GAL* zu verdanken ist. Bei diesem *GAL* handelt es sich um einen programmierbaren Logikbaustein, der hier praktisch die Rolle der 'allesverbindenden Logik' spielt. Durch diese Maßnahme konnten etwa 5 weitere Logik-Bausteine eingespart werden.

Video und EASYTIZER

Zur Sache: betrachtet man zunächst einmal den Analog-Teil des Easytizers, so wird man wohl sinnvollerweise bei der Video-Eingangsbuchse anfangen. Zuerst gelangt das Video-Signal vom Eingang zu einem Emmitterfolger (T1) mit hohem Eingangswiderstand und niedrigem Ausgangswiderstand, wodurch eine Rückwirkung der Schaltung (durch die Komparatoren usw.) ausgeschlossen wird. Danach läßt sich die Schaltung in 3 Funktionsbereiche aufteilen:

1.) Synchronsignalabtrennung und „Klemmschaltung“

Die Synchronabtrennstufe inklusive Klemmschaltung ordnet sich rund um einen Komparator aus IC10 an. Nachdem das Videosignal durch einen Tiefpaß entstört und erneut gepuffert wurde, gelangt es zur eigentlichen Synchronabtrennung. Da der Synchronpegel der niedrigste des ganzen Videosignals ist, wird sich der Synchronpegel an dem Knotenpunkt C3/D1 auf die Spannung am Knotenpunkt R14/C4 (etwa 3.2V) minus der Durchbruchspannung von D1 und minus dem Spannungsabfall über R9 (etwa 100mV SS) einstellen. Die zum Zeitpunkt des Synchronimpulses an R9 abfallende Spannung wird zum einen von dem Strom durch R8 und dem konstanten Tastverhältnis (Synchron/Rest) bsetimmt. Der Strom durch R8 bleibt weitestgehend signalunabhängig und damit auch die Synchronamplitude über R9. Ein nachfolgender Komparator (IC10) wertet diesen Spannungsabfall über R9 aus und springt mit seinem Ausgang auf 1, sobald die Spannung am Knotenpunkt D1/R9 die am nichtinvertierenden Eingang anliegende Spannung unterschreitet. R15 und C8 dienen lediglich der Störsicherheit: Somit liegt das positive Synchronsignal am Ausgang des Komparators an. Dieses Synchronsignal gelangt über ein Differenzierglied an das Gate des selbstsperrenden MOSFET T5. Durch das kurzzeitige Leiten von T5 bewirkt dies ein 'Festklemmen' des Synchronpegels und damit Bezugspegels auf genau 0V. Das ist nötig, da Videosignale in den meisten Fällen gleichspannungsfrei übertragen werden und der Weißbeziehungsweise Schwarzwert von der Signalamplitude abhängt. Klemmt man jedoch das Videosignal am Synchronpegel fest, so ist die Helligkeitsinfor-

ROM-PORT

IC10..IC11 = LM319

5V ca. 150mA

Gesamtschaltbild des Easytizers

Easytizer Stückliste

Halbleiter
D1: 1N4148
T1, T2, T6: BC548C o. ä.
T3, T4: BC558C o. ä.
T5: BS170
IC1, IC2, IC3, IC4: 74HC595
IC5: 74HCT138
IC6: 74HC393 oder 74HCT393
IC7: 74HC191
IC8: GAL16V8-35 (programmiert)
IC9: ICL7660
IC10, IC11: LM319

Widerstände
alle 1/4 W Kohleschicht
R1: 75Ω
R2: 27K
R3: 82K
R4, R23: 220Ω
R5, R7, R9, R26, R35: 1K
R6, R22: 10K
R8: 1M
R10: 100Ω
R11: 15K
R12, R21: 5K6
R13: 1M5
R14: 8K2
R15, R32: 22K
R16: 390Ω
R17: 1K8
R18, R19: 3K3
R24: 2Ω2
R25: 10Ω
R27, R28, R29: 470Ω
R30: 12K
R31: 47K
R33, R34, R36, R37: 4K7
P1: Potentiometer 1K 4mm Achse
P2: Trimmer 1K klein liegend

Kondensatoren
C1: 47µ / 16V Elko stehend
C2, C4, C6, C12, C13, C15: 10µ / 16V Elko stehend
C3: 1µ Folie
C5: 2µ2 / 16V Elko stehend
C7: 330p Keramik
C8: 47p Keramik
C10: 10n
C14: 220µ / 16V Elko stehend
C16: 100µ / 16V Elko stehend
C17, C28, C19, C20, C35: 100n mini
C21: 4n7 Folie
C22: 1n2 Folie
C23: 1n Folie
C24: 100p Keramik
C25 bis C32: 8 mal 100n mini
C33: 10µ / 16V Tantal
C34: 100µ / 16V Elko stehend

Sonstiges
Q1: 32Mhz Quarzoszillator CMOS
BU1, BU2: Cynchbuchse für Printmontage
ST1, S1, J1: 8-polige Pfostenleiste
1 Platine 005 – Easytizer
1 Adressierstecker
4 Abstandsbolzen M3★15
4 Muttern M3
3 Stunden Geduld
S1 mini Kippschalter 1 mal Um

Für das Trigger-Kabel:
bis zu 10m einadrig abgeschirmtes Kabel
1 Taste (Schließer)
1 Cynch-Stecker

Bestückungsplan (vergrößert)

mation nicht mehr relativ zum Synchronpegel des Videosignals, sondern fest auf 0V bezogen.

2.) Die Bildsignalauswertung

Das nun 'Festgeklemmte' Videosignal gelangt an die nichtinvertierenden Eingänge von 3 Komparatoren, deren invertierende Eingänge wiederum mit einer als 'Referenzleiter' dienenden Spannungsteilerkette verbunden sind. Die mit P2 einstellbare Spannung am unteren Ende des Spannungsteilers bestimmt den Schwarzwert und die am oberen logischerweise den Weißwert. Mit diesen 3 Komparatoren ist ein sogenannter Flash-A/D-Wandler realisiert, der an den Ausgängen den digitalen Momentwert des analogen Eingangssignals dezimal codiert hervorbringt. Die Anzahl der Komparatoren, deren Ausgänge logisch high werden, ist ein Maß für die Eingangsspannung (Helligkeit) und somit für eine der vier Graustufen. Sind alle Ausgänge null, so entspricht dies dem niedrigsten Grauwert (schwarz). Diese dezimal codierte Information wird dann im nachfolgenden GAL zu einem 2-Bit Binär-Code decodiert.

3.) Der Weißwertgleichrichter u. die negative Spannungsversorgung

Der um T2 und T4 aufgebaute Weißwertgleichrichter dient lediglich dazu, den Maximalwert des Videosignals und damit die hellste Stelle festzuhalten, um ihn dann wahlweise als Referenz der oberen Spannungsteilerkette heranzuziehen (Automatische Helligkeitseinstellung).

Die zum Betrieb des Analog-Teils nötige negative Spannung wird ganz konventionell mit IC9 erzeugt. Außerdem sind die Versorgungsspannungen des Analogteils aufgrund der möglichen Störeinkoppelungen jeweils durch ein R/C-Glied entkoppelt.

Der Digitalteil

Um ein Bild zu Digitalisieren, muß man sowohl das Zeitraster (Takt) als auch die Zahl der analogen Werte (Graustufen) in eine begrenzte Anzahl von endlich kleinen Werten aufteilen. Der Punkttakt beträgt beim Easytizer exakt 16 MHz. Das bedeutet: mit einer Frequenz von 16 MHz wird dem Videosignal eine Probe entnommen, die Auskunft über die Helligkeit in vier Stufen gibt. Diese zwei Bit werden dann mit diesem Takt von 16 MHz in jeweils zwei 16-Bit Schieberegister (IC1/IC3, IC2/IC4) geschoben. Den Takt für die Schieberegister liefert der Ausgang Qa der einen Hälfte des zweifachen 4-Bit Binärzählers IC6, der seinerseits von einem 32-MHz Quarzoszillator getaktet wird. Dieser Zähler wird von dem Synchronsignal gelöscht und damit synchronisiert. Ausgang Qc dieses Zählers taktet wiederum den synchronen 4-Bit Binärzähler IC7. In IC7 wird mit dem Synchronsignal der Zählerstand der anderen Hälfte von IC6 geladen, der allein der Softwarekontrolle unterliegt. Von IC7 wird lediglich der Übertragausgang (/CRY) benutzt, welcher bei einem Überlauf von IC7 den Schieberegister-

Oszillogramm 1: Zeile eines FBAS-Video-Signals. (Erkenntlich am Farb-Burst direkt hinter dem Synchronsignal.)
Horizontal: 10 µs/cm
Vertikal: 0,2V/cm

Oszillogramm 2: Das abgetrennte Zeilensynchronsignal
Horizontal: 10 µs/cm
Vertikal: 2V/cm

inhalt von IC1–IC4 in die intern nachgeschalteten Register überträgt. Das geschieht aber nicht periodisch alle 16 Bittakte, sondern nur alle 64 und außerdem abhängig von dem Dateneingang n (an IC7) in n★4 Stufen versetzt. Grund für diese Maßnahme ist, daß die Datenübertragung zum Rechner hin nicht mit 32 Mbit/s erledigt werden kann, sondern höchstens mit 8 Mbit/s. Also wird eine Videozeile in Spalten zu je sechzehn Bildpunkten eingeteilt, die sich im Abstand zu 64 Bildpunkten wiederholen. Um trotzdem alle Spalten für den Rechner zugänglich zu machen, ist die Verschiebung der Spalten in Schritten zu 16 Bildpunkten über den Zählerstand der zweiten Hälfte von IC6 softwaremäßig einstellbar. Damit wurde die nötige Übertragungsrate auf ein viertel reduziert, der Zeitaufwand ist jedoch um das vierfache gestiegen (80ms statt 20ms). Ein weiterer Nachteil soll nicht verschwiegen werden: Durch das Zusammenfassen von 4 aufeinanderfolgenden Bildern zu einem Bild im Rechner sollte sich der Bildinhalt über vier Perioden hinweg so gut wie nicht verändern, da es sonst zu störenden Verschiebungen kommen kann. In der Realität hat sich jedoch gezeigt, daß sich bewegte Bilder doch noch recht gut digitalisieren lassen, da eine Digitalisierzeit von 80 Millisekunden vergleichsweise kurz ist.

Oszillogramm 3: Oben (3a): Die Rasterwechselimpulsfolge des Synchronsignals
Unten (3b): Das abgetrennte, vertikale Synchronsignal. (Bildfrequenz, 50Hz)
Horizontal: 100 µs/cm
Vertikal: 2V/cm

Die Seriell-parallel-Wandlung

Die Dateneingänge der Schieberegister sind mit den Ausgängen von IC8 verbunden, die die decodierte Information der 3 Komparatoren liefern. Die Ausgänge der Schieberegister sind direkt mit dem Datenbus des Rechners verbunden und werden über den 1 aus 8 Decoder IC5 aktiviert. Der Rest des Digitalteils setzt sich aus IC5, IC8 und wenigen passiven Bauelementen zusammen. IC5 steuert den Zugriff auf die Hardware-Ressourcen in Abhängigkeit von der Zugriffsadresse und den Steuersignalen /UDS und /ROM3. IC8 bildet außer dem bereits erwähnten Decoder auch ein durch den Rechner lesbares Statusregister, welches Informationen über den Zustand der Hardware gibt. Die an IC8 angeordneten passiven Bauelemente dienen lediglich einer Verlängerung des Synchronimpulses beziehungsweise dem Abtrennen der Bildfrequenz aus dem Synchrongemisch.

Die besondere Rolle des GAL's

Bei diesen GAL's handelt es sich wie Sie vielleicht schon wissen um eine weiterentwickelte Variante der PAL-Familie. Gegenüber PAL-Bausteinen sind GAL's in CMOS-Technologie gefertigt, zeigen einen geringeren Leistungsverbrauch, sind in den dynamischen Eigenschaften überlegen und besitzen eine wesentlich flexiblere Ausgangsstruktur. Durch den Einsatz dieses GAL's konnte die nötige Gatterlogik auf null reduziert werden. Die Funktionen, die das GAL hier erfüllt, sollen im Folgenden kurz zusammengefaßt werden.

1.) Decodieren der von den 3 Komparatoren gelieferten Information zu einem 2-Bit Binärcode.

2.) Bildung eines vom Rechner lesbaren Statusregisters, das über Synchronsignale, die Schnappschusstaste und den Zustand der Schieberegister (voll/leer) Auskunft gibt.

Abbildung 2 zeigt die Jedec-Datei, welche zum Konfigurieren des GAL's benötigt wird. Diese Datei muß in ein entsprechendes Programmiergerät übertragen werden. Falls Sie sich nicht in der Lage sehen, diesen Baustein mit Hilfe der von uns gelieferten Informationen zu programmieren, steht es ihnen frei, das GAL 'EASYTIZER' von der Redaktion zu beziehen.

Aus hart mach weich...

Und nun zur softwareseitigen Ansteuerung der EASYTIZER-Hardware am ROM-PORT des ST. Der ROM-Bereich \$FB0000–\$FBFFFF wird von den Registern des EASYTIZER's belegt. Es sind die 8 Register, für die nur **lesender** Wortzugriff oder lesender Byte-Zugriff auf geraden Adressen zugelassen ist:

\$FB0000 SR0, nur Wort:
Niederwertigster Teil der Bildinformation, aus den mit dem Spaltenzähler eingestellten Spalten, der sich alle 4 Microsekunden verändert. Ein Zugriff auf diese Adresse löscht gleichzeitig die LWF-Flagge im Statusregister.

\$FB0002 SR1, nur Wort:
Höchstwertiger Teil der Bildinformation, aus den mit dem Spaltenzähler eingestellten Spalten, der sich alle 4

```
JEDEC file for: P16V8C
Gemischte Logik des Easytizers
in ein GAL 16V8 integriert
Raymund Hofmann 16/7/87*
QP20* QF2194*
L0000
11111111111111111111111111111111
01111111011111111111111111111111
00000000000000000000000000000000
00000000000000000000000000000000
00000000000000000000000000000000
00000000000000000000000000000000
00000000000000000000000000000000
00000000000000000000000000000000
11111111111111111111111111111111
10100111011111111111111111111111
01101011011111111111111111111111
10011011011111111111111111111111
01010111011111111111111111111111
00000000000000000000000000000000
00000000000000000000000000000000
00000000000000000000000000000000
11111111111111111111111111111111
11111110111111111111111111111111
00000000000000000000000000000000
00000000000000000000000000000000
00000000000000000000000000000000
00000000000000000000000000000000
00000000000000000000000000000000
00000000000000000000000000000000
11111111111111111111111111111111
11111111011110111110111111111111
11111111011111111111011110111111
00000000000000000000000000000000
00000000000000000000000000000000
00000000000000000000000000000000
00000000000000000000000000000000
00000000000000000000000000000000
11111111111111111111111111111011
11111111111110111111111111111111
00000000000000000000000000000000
00000000000000000000000000000000
```

```
00000000000000000000000000000000
00000000000000000000000000000000
00000000000000000000000000000000
00000000000000000000000000000000
11111111111111111111111111111011
11111110111111111111111111111111
11111111111111111111111111111110
00000000000000000000000000000000
00000000000000000000000000000000
00000000000000000000000000000000
00000000000000000000000000000000
00000000000000000000000000000000
11111111111111111111111111111011
11111111111101111111111111111111
00000000000000000000000000000000
00000000000000000000000000000000
00000000000000000000000000000000
00000000000000000000000000000000
00000000000000000000000000000000
00000000000000000000000000000000
11111111111111111111111111111011
11111111111111110111111111111111
00000000000000000000000000000000
00000000000000000000000000000000
00000000000000000000000000000000
00000000000000000000000000000000
00000000000000000000000000000000
00000000000000000000000000000000*
L2048
11011110*
L2056
01100100011010010101001001001000
00110100001110000011011100100000*
L2120
11111111*
L2128
11111111111111111111111111111111
11111111111111111111111111111111*
L2192
11*
C5D2F*
03FF
```

Abbildung 2: Die Jedec-Datei enthält alle Informationen um das GAL zu Konfigurieren. Über die Redaktion kann der Quellcode im ABEL-Format bezogen werden.

Microsecunden verändert. Anmerkung: diese Register sind so organisiert, daß sie direkt mit den zwei Bildebenen in mittlerer Auflösung korrespondieren. (Also ist Bit 15 der am weitesten links liegende Bildpunkt.)

$FB0004 INZ, Wort/Byte:
Ein Lesezugriff auf dieses Register erhöht den Spaltenzähler um eins. Die gelesenen Daten sind irrelevant.

$FB0006 LZ, Wort/Byte:
Ein Lesezugriff auf dieses Register setzt den Spaltenzähler auf Null zurück. Die gelesenen Daten sind irrelevant.

$FB0008 STATUR, Wort/Byte:
Status-Register, das über verschiedene Signale im EASYTIZER Auskunft gibt. Von 16 Bits werden lediglich 4 benutzt:

Bit 15: LWF
Diese Flagge wird gesetzt, wenn ein neues Langwort aus den Schieberegistern IC1 – IC4 in die intern nachgeschalteten Register übertragen wurde und wird durch einen Zugriff auf SR0 wieder gelöscht. Zweck dieser Flagge ist es, eine Synchronisation zwischen Rechner und Videosignal am Anfang einer Zeile zu ermöglichen.

Bit 14: CSYNC
Über ein Monoflop um etwa 7,8 Microsecunden verlängertes, positives Synchronsignal, das ebenfalls zur Synchronisation des Rechners mit dem Videosignal dient.

Bit 13: VSYNC
Dieses Bit wird für etwa 160 Microsekunden Dauer während eines Rasterwechsels aktiv.

Bit 12: TASTE
Dieses Bit wird bei gedrückter 'Schnappschuß-Taste' aktiv. Die Schnappschuß-Taste wird über BU2 an den Easytizer angeschlossen.

Schlußbemerkung

Im zweiten und letzten Teil dieser Serie geht es dann um die in Assembler verfassten Digitalisierroutinen und das genaue Software-Protokoll zur Ansteuerung des EASYTIZIER's. Dadurch ist es möglich, diese Routinen in selbst verfasste Programme einzubinden und sich somit eine eigene Benutzeroberfläche zu programmieren. Weiterhin wird die fertig erhältliche Software beschrieben. Das fertig programmierte GAL und die Platine sowie ein Fertiggerät kann über die Redaktion bezogen werden.

(RH)

A-NET

– ein preiswertes lokales Netzwerk für den Atari

Mit dem immer größeren Wunsch, Daten zentral zu erfassen und allgemein zugänglich zu machen, gewinnen Daten-Netzwerke immer mehr an Bedeutung. Es gibt für sie viele Konzepte, die alle ihre Vor- und Nachteile haben. Prinzipiell kann man diese Netzwerke von ihrem Aufbau her in mehrere Gruppen einteilen. Zu diesen Gruppen gehört das sternförmige Netz, die „Party-Line" und das ringförmige Netz. Abgesehen von diesen Grundformen, lassen sich natürlich auch einige sinnvolle Kombinationen zusammenstellen, um ein leistungsfähiges Netzwerk für bestimmte Anwendungen zu erhalten. Die gewählte Netzform ist meist abhängig von den räumlichen Standorten der Rechner.

Master-Slave

Ein relativ einfach zu realisierendes Verfahren ist das des Master und Slave. Ein Rechner in dem Netzwerk fungiert als Master und die anderen als Slave. Der Master hat die Aufgabe, das Netzwerk zu verwalten. Eine Anforderung eines Slave an den Master kann prinzipiell auf zwei Arten erfolgen. Die Erste nennt sich „Polling". Der Master fragt der Reihe nach die angeschlossenen Slaves ab, ob sie bedient werden wollen. Die zweite Art läuft über „Interrupt". Dabei meldet sich der Slave, wenn er eine Bedienung wünscht, über einen Interrupt bei dem Master. Wenn allerdings der Master mit einem anderen Rechner gerade kommuniziert, muß die Anforderung über eine getrennt Leitung erfolgen, da zur Zeit des Auftretens einer Anforderung Daten über das Netz ausgetauscht werden können.

Aufwender und wesentlich teurer sind Verfahren, in welchen jeder Teilnehmer am Netz mit der gleichen Berechtigung arbeitet. So kann (in einem gleichberechtigten Netz) jeder mit jedem kommunizieren.

A-NET

Die uns vorgestellte Version von A-NET benutzt den Aufbau des ringförmigen Netzes. Dazu bekommt jeder Teilnehmer eine kleine Box für den Netzwerkanschluß, die an die Midi-Schnittstelle angeschlossen wird. Die Verbindungen zwischen den Boxen stellt ein LWL (Licht-Wellen-Leiter) her. Diese moderne Technik erlaubt einen störungsfreien Betrieb des Netzes. Die Daten werden mit einer Übertragungsrate von 31 250 Baud über das Netz ausgetauscht. Die maximle Länge des Lichtwellenleiters zwischen zwei Rechnern kann ohne Verstärker 40 Meter betragen. Das Netzwerk läßt sich wie ein weiteres Laufwerk mit der Kennung „N" ansprechen. Durch die ringförmige Struktur des Netzes muß jede Interface-Box aktiv sein, damit das Netz arbeitet und somit der Ring geschlossen ist. Die Interface-Box wird entweder über den Joystick-Port oder ein externes Netzteil mit Spannung versorgt. Es empfiehlt sich aber die letztere Lösung, damit nicht jeder Rechner wegen der Spannungsversorgung der Interface-Box andauernd eingeschaltet sein muß. Denn ist auch nur eine Box inaktiv, ist der Ring nicht mehr geschlossen und der Datentransfer nicht mehr möglich.

In der Grundausstattung arbeitet der Master als reiner Fileserver, d. h. er dient zur reinen Verwaltung der Dateien und kann vom Anwender nicht direkt genutzt werden. Jeder Slave hat die Möglichkeit, über das Netzwerk mit der Partition „D" des Masters zu arbeiten. Ihnen stehen alle Files und Daten auf dieser Partition zur Verfügung. In der jetzigen Version wird ein Slave erst vollständig abgearbeitet. Dies bedeutet, daß jeder Slave, der in dieser Zeit eine Anfrage an den Master startet, solange warten muß, bis der Master wieder frei ist. Jedoch wird dieses Manko in der nächsten Version des A-NET beseitigt sein. Dies wird durch eine Blockübertragung erreicht werden, die dann auch eine Blocksicherung der Daten zuläßt, obwohl es sich bis jetzt als nicht notwendig erwies. Mit dieser Maßnahme können Anforderungen, wie z. B. der Aufruf eines Directories oder das Lesen eines Datensatzes, ausgeführt werden, während ein Anderer gerade ein langes File lädt.

Weitere Probleme treten bei Datenverarbeitungsprogrammen auf, da sie für den Einsatz als „Stand alone" konzipiert sind. Der Betrieb ist solange problemlos, solange nur einer mit einer Datei arbeitet. Wollen zwei Anwender die gleiche Datei bearbeiten, so gilt die zuletzt abgespeicherte Version. Daten die vorher bearbeitet wurden, sind verloren. Noch schwieriger wird es bei indexsequentiellen Dateien, da dann die Index Datei bei einer Einfü-

gung eines Datensatzes neu organisiert wird. Dies kann dann zu Problemen führen, wenn eine zweite Person in diesem Datensatz weiterblättert. Auch hier werden in der neusten Version einige Verbesserungen eingebaut sein. Die Lösung lautet „File Locking“. Wird eine Datei bearbeitet, so ist der Zugriff anderer Rechner auf diese Daten unterbunden.

Bindings

Für den professionellen Einsatz von Datenverarbeitungsprogrammen stellt die Firma einige Bindings gegen Entgelt zur Verfügung. Die Bindings enthalten alle Routinen für eine indexsequentielle Dateiverwaltung im Multiuser-Betrieb. Die Bindings werden für alle gängigen Sprachen geliefert. Damit lassen sich Datenverarbeitungsprogramme im Multiuser-Betrieb entwickeln, die dann mit A-NET lauffähig sind.

Außerdem bekommt man noch 50 Runtime Lizenzen, um das mit diesen Routinen entwickelte Produkt verkaufen zu können. Für Fragen steht eine Hotline zur Verfügung, damit man sich nicht alle Haare ausrauft, wenn es mal zu Problemen kommt.

Utilities

Mit einem Drucker- und Spooler-Accessory lassen sich Daten problemlos über das Netzwerk ausdrucken. Um das Netzwerk wegen des langen Druckvorgangs nicht zu sehr zu blokkieren, ist es ratsam, einen weiteren Druckerspooler im Master zu installieren. Dadurch stellt der Druckvorgang eine minimale Belastung für das Netzwerk dar.

Mit der Systemzeit bekommen alle Slavs eine Uhrzeit vom Master, die dann im Atari gesetzt wird.

Update

Die Firma DM Computer GmbH bietet einen Updateservice an, der jeden Käufer dieses Produktes auf den neusten Stand bringt. Die neueste Version wird voraussichtlich in ein bis zwei Monaten auf dem Markt erhältlich sein.

Dieses Netzwerk ist schon in mehreren Firmen installiert. Dabei zeigte sich ein sinnvolles Arbeiten mit maximal acht Teilnehmern, obwohl bis zu 255 Teilnehmer möglich wären. Mit den genannten Verbesserungen in der neusten Version wird sich die sinnvolle Teilnehmerzahl erhöhen.

Das Angebot

Das Komplettsystem beinhaltet Interfaces für ein Master und zwei Slaves, Software und dreimal fünf Meter Lichtwellenleiter und kostet DM 1450,–. Jedes weitere Interface für ein Slave mit Software und fünf Meter Lichtwellenleit er kostet DM 450,–. Es besteht auch die Möglichkeit, längere Lichtwellenleiter zu beziehen, wobei ein Meterpreis von DM 8,– zugrundeliegt. Ebenfalls sind externe Netzteile zum Preis von DM 35,– zu beziehen.

Als zusätzliche Software sind das Drucker- und Spooler-Accessory (1x pro Netz) für DM 450,–, sowie automatische Systemzeit für Dm 75,– erhältlich. Für die Programmierer gibt es die Binding smit 50 Runtime Lizenzen für DM 4500,–.

Sven Schuler

DM Computer GmbH
Durlacherstr. 39
7530 Pforzheim
Tel. 07231/13939
Telex 783248

Aus der Hardwareküche der ST

Der Easyprommer

Ein universelles Programmiergerät für den ATARI ST, das nicht nur alle gängigen EPROMs der 27...-Serie brennt, sondern auch die modernen EEPROMs lesen und programmieren kann. Die sehr komfortable Software, natürlich voll GEM unterstützt, erlaubt alle nur denkbaren Manipulationen. Mit dem integrierten HEX/ASCII-Monitor lassen sich Änderungen der Daten blitzschnell realisieren. Der auf der Platine befindliche Druckertreiber erlaubt das problemlose Drucken der Daten. Zwei Zusatzprogramme ermöglichen das Brennen **jeder** nicht kopiergeschützten Software direkt von Diskette einschließlich Accessories und Autostart-Programmen. Der Easyprommer wird an den Druckerport angeschlossen und samt Diskette und Handbuch geliefert.

Die 576 KByte Easybank

Diese 'gebankte' EPROM-Karte erlaubt es den ROM-Speicher des ATARI ST um über **570 Kilobyte** zu erweitern. Damit haben Sie die Möglichkeit ganze Programmpakete sofort nach dem Einschalten des Rechners parat zu haben. Selbst das Booten von Autostart-Programmen und Accessories ist möglich. Auf der Easybank finden bis zu acht EPROMs des Typs 27512 oder 27256 Platz, wobei die Steuersoftware automatisch in zwei weitere EPROMs des Typs 27256 mitgebrannt wird. Die Platine wird mit vergoldeten Anschlüssen samt Diskette, die die Software zum erzeugen der EPROM-Files enthält, sowie ausführlicher Bedienungsanleitung geliefert.

Die 128 KByte EPROM-Karte

Diese 'normale' EPROM-Karte kann bis zu vier EPROMs des Typs 27256 oder 27128 oder 2764 aufnehmen. Sie eignet sich somit für kleinere Programme.

Verlängerungskabel zum Easyprommer

Mit diesem Kabel kann der Easyprommer auch neben dem Rechner betrieben werden. Dieses Kabel ist notwendig, wenn der Easyprommer im Gehäuse geliefert wird.

Neue Softwareversion zum Easyprommer!

Ab sofort wird der Easyprommer mit der neuen Software V.2.2 ausgeliefert. Neben kleineren Verbesserungen ist vor allem das neue Megabit EPROM 27011 implementiert worden. Ferner werden Fehlermeldungen beim Programmieren, Vergleichen ect. jetzt mit der fehlerhaften Adresse ausgegeeben. Kunden, die eine ältere Version besitzen, erhalten bei Einsendung der Original-Diskette und eiens frankierten Rückumschlages sowie eines Kostenbeitrages von DM 10,– die neue Version V.2.2.

Hiermit bestelle ich:

- ☐ Easyprommer (Fertiggerät) für DM 349,–
- ☐ Easyprommer (Bausatz) für DM 298,–
- ☐ 576 KByte EPROM-Karte (fertig bestückt o. EPROMs) für DM 148,–
- ☐ 128 KByte EPROM-Karte (fertig bestückt o. EPROMs) für DM 58,–
- ☐ Verlängerungskabel für Easyprommer für DM 39,90

Versandkosten: Inland DM 7,50 Ausland DM 10,– ☐ Vorauskasse
Auslandsbestellungen nur gegen Vorauskasse ☐ Nachnahme

Name: ______ Vorname: ______

Straße: ______ Ort: ______

Unterschrift: ______

Bezugsquelle: MERLIN Computer GmbH
Industriestr. 26
6236 Eschborn
Tel.: 0 61 96 / 48 18 11

Neu für den ATARI-ST, aber auf Großrechnern, Minis und Supermicros (Workstations) weit verbreitet: APL. Hier ist der Test. Zur Einführung wird der Artikel in der ST 9/87 empfohlen, der die Programmiersprache APL allgemein vorstellt.

Ich halte es für eine gute (anglo-amerikanische) Sitte, zu Beginn des Artikels für zwei Klarstellungen zu sorgen: Zum Einen verdiene ich mit APL-Programmen seit 9 Jahren meine Brötchen und bin überzeugt, daß ohne APL die Brötchen kleiner ausfallen würden – ich bin also voreingenommen. Zum Anderen war ich zur Erstellung eines gründlichen Tests auf die Hilfe des deutschen Distributors angewiesen: Die gdat-mbH in Bielefeld war zum Beispiel als APL-Systemhaus so freundlich, die APL-Einschübe zu setzen. – Zeitschriftenverlage tun sich mit den Spezialzeichen schwer. Auch eine solche Kooperation könnte als Befangenheit ausgelegt werden. Andererseits sind von gdat Hinweise auf Fehler gekommen, die bei einem Test gemeinhin nicht auffallen. Zudem danke ich für die vergleichenden Benchmarks und für Hintergrundinformationen.

Das bisher einzige APL für den ATARI-ST heißt APL/68000. Dieses APL wird seit 1981 auf immer neuen Rechnern implementiert, denen nur eines gemeinsam ist: Eine CPU aus der 680xx-Serie von Motorola. Die Herstellerfirma MicroAPL in London hat bisher Anpassungen für Betriebssysteme wie UNIX, CPM/68k, TOS und MIRAGE, für Workstations wie die SUN, für Mehrplatzrechner (z. B. Hewlett Packard) und für „WIMP"-Rechner (Windows, Icons, Mäuse, Pulldown-Menüs) vorgenommen: APL/68000 gibt es u. a. für den MAC, die AMIGA und für den ST. Der „Rumpf" des APL/68000 ist bei allen Maschinen weitgehend identisch, egal ob die Betriebssysteme multitasking-fähig sind oder nicht (das läßt für den ST hoffen). Lediglich das „Interface" zur Außenwelt wird aktuell angepaßt.

Getestet wurde die Version 6.00A (Erstversion für den ATARI) sowie ein Vorabexemplar der Version 6.05. Letztere Version ist bei Erscheinen des Artikels nach Angaben des Distributors im Handel. Da ich von der ST-Computer in keinster Weise abhängig bin, darf ich sie wohl mal loben: Ich konnte APL/68000 ein halbes Jahr testen. Sie lesen also gewiß keinen Schnellschuß, sondern einen gründlichen Test. Dies ist einer so komplexen Programmiersprache wie APL nur angemessen.

„Speedy Gonzales"

Das APL/68000-ST kommt in einem professionell aufgemachten Schuber. Das Handbuch (ein Ringbinder im Format DIN-A5) ist in Englisch gehalten, kommt aber ohne „Fachchinesisch" aus und ist hervorragend geschrieben. Es werden nicht nur alle Sprachelemente (mit Beispielen!) beschrieben, auch ein Index und ein Tutorial (sehr empfehlenswert) sind enthalten. Zusätzlich gibt es ein Manual für die ATARI-Implementation, worin die Besonderheiten der ST-Version beschrieben sind. Liebe zum Detail zeigt sich bei einer Kurzreferenz, die beim Programmieren immer dabei sein kann, ohne der Maus den Platz streitig zu machen. Eine (deutschsprachige) Einführung in die Besonderheiten von APL komplettiert das Paket. Eine (einseitige) Diskette mit dem APL gibt's natürlich auch. Kein Kopierschutz oder ähnliche Bosheiten. Zudem legt gdat in der BRD eine eigene Disk bei, die einige Tools enthält. Besonders nützlich: der Feld-Editor für Anwenderprogramme.

Zusätzlich standen mir zwei Software-Pakete von gdat zur Verfügung: Ein Druckertreiber (APLPRINT) und ein Grafik-Workspace (LineAl). Diese Tools sind zwar bei der Arbeit mit APL ausgesprochen hilfreich, können aber in diesem Test aus Platzgründen nicht weiter berücksichtigt werden.

„Echte" APLer hassen Handbücher. Sie legen einfach los. Diskette einschieben und APL.PRG angeklickt – kur-

ze Ladezeit – voil: Eine GEM-Applikation. Mit eigenem Fenster, Menübalken usw. – Schön! Erste Tests mit Programmen, die ich vom IBM-PC übernommen habe, frustrieren mich: War's das schon? Das Ding ist entweder unglaublich schnell, oder es simuliert nur. Na warte. Der ST hat keinen Mathe-Chip (im Gegensatz zu meinem alten PC). Bei Fließkomma-Arithmetik in gewohnter APL-Präzision kommt der Bursche bestimmt in's Schwitzen? – Pech gehabt. Das Biest rennt wie verrückt. Eine Stoppuhr muß her. Tausend Integers sortieren. Ich tippe die fünf Zeichen ein, drücke RETURN – Fertig! Mein Daumen war nicht schnell genug. Ich kombiniere: APL/68000 ist blitzschnell.

Die Benchmarks bestätigen diesen Eindruck. Nicht nur, daß APL (unter Berücksichtigung der höheren Rechengenauigkeit) mit dem GFA-Basic mithalten kann. Ein solcher Vergleich hinkt immer, wenn man an die vielen Sprachelemente denkt, die in keiner anderen Sprache eingebaut sind. Nein, der wirkliche Hammer ist, daß es APL auf dem ST mit einer reinen Softwarelösung mit dem Arithmetik-Prozessor des PC aufnehmen kann. Das zeugt von sorgfältiger Codierung. Nur bei der Potenz-Funktion muß jemand bei MicroAPL einen schlechten Tag gehabt haben.

Technische Daten des APL/68000-ST

Verfügbarer Arbeitsspeicher (⎕WA): (bei 1040ST. Allg.: RAM abzüglich ca. 275k)	725k
Numerischer Darstellungsbereich: von bis	 ¯3.59538626972463 E 308 3.59538626972463 E 308
Darstellgenauigkeit (⎕PP):	≤15 Nachkommastellen
Vergleichstoleranz, relativ (⎕CT):	≥1E¯307
Genauigkeit des Timers (⎕TS):	20 msek.
Datentypen:	Character (8 Bit) Boolean (1 Bit) Integer (32 Bit) Reals (IEEE, 64 Bit)
Größe von Variablen: nur durch verfügbaren Arbeitsspeicher begrenzt	
Anzahl der Dimensionen:	maximal 8
Größe einer Dimension:	unbegrenzt
Rekursionstiefe (Stack): nur durch verfügbaren Arbeitsspeicher begrenzt	
Symbol-Table:	maximal 6021 Einträge
APL.PRG V.6.05 (auf Disk):	118k
APL.RSC (auf Disk):	3,5k
Belegter Diskspace (incl. Bibliotheken):	275,8k
Betriebsarten:	monochrom (640 × 400) 4-farbig (640 × 200) 16-farbig (320 × 200)

Abb. 1

Dem Paket liegt auch eine Fehlerliste bei. Angesichts dessen, daß es kaum fehlerfreie Programme gibt, ist diese Praxis lobenswert. Da gdat an alle registrierten Benutzer die Version 6.05 verschickt, sobald sie freigegeben ist, erhält jeder APL-Eigner eine komplette Fehlerkorrektur und einige neue Features: gegen Porto. So ein Nulltarif ist angesichts der meist kleineren Bugs sehr nobel. Andere Softwarehäuser sollten sich daran ein Beispiel nehmen. Der einzige Bug, der mich nachhaltig nervte, betraf einen gelegentlichen Ausstieg des Rechners bei dem Versuch, einen Workspace zu laden.

Bombensicher

Apropos Bomben: Der ATARI ist ja in dieser Disziplin ein echter Anarchist. Aber außer in dem eben beschriebenen Fall konnte ich unter APL keinerlei Abstürze provozieren. APL ist von Haus aus so konzipiert, daß ein Benutzer von den Ärgernissen des Betriebssystems abgeschirmt bleibt. APL/68000 unter GEM bildet da keine Ausnahme. Ich bin bezüglich dessen, was der ST gemeinhin bei der Programmentwicklung an Tretminen bereithält, regelrecht fahrlässig geworden. APL/68000 hält das aus. Erste Versuche mit Version 6.05 zeigten absolut keine Fehler mehr. Und das will was heißen: Ich habe immerhin einige hundert Kilobytes an Programmen auf dem ST unter APL laufen lassen.

Per Mausklick ist auch ein Editor erreichbar, der für die üblicherweise sehr kurzen APL-Funktionen (Module) mehr als ausreichend ist. Die „CCP"-Fähigkeiten (Cut, Copy, Paste) sind sehr hilfreich. Leider lassen sich mit diesem Editor keine Variablen bearbeiten. Diese Fähigkeit hatte ich auf einem APL für PC's doch sehr zu schätzen gelernt. Zwar ist eine diesbezügliche Erweiterung geplant, aber ein Termin steht laut **gdat** noch nicht fest.

Die APL-Befehle bestehen aus „Icons", die auf älteren Systemen oft nur schwer zu erreichen waren: Tastatur und Bildschirm gaben oft nur die herkömmlichen ASCII-Zeichen her. Schon bei den Umlauten wurde es schwierig.

Der ST zeigt sich hier von seiner besten Seite. Alle APL-Zeichen sind auf dem Bildschirm darstellbar, die Tastatur läßt sich (mit CONTROL/TAB oder unter Programmkontrolle) zwischen der vom PC bekannten APL-Tastatur und dem (deutschen) Tastenlayout umschalten. Der APL-Zeichensatz enthält sämtliche europäischen Sonderzeichen, das komplette ASCII-Repertoire und viele ATARI-Symbole (wer hat eigentlich schon mal das „Pfeifenmännchen" gebraucht?). Die APL-Belegung läßt sich mit Tastaturaufklebern markieren. Sie gehören zum Lieferumfang. In der neuen Version kann man sogar beide Zeichensätze (16x8; 8x8) als Bitmaps lesen und verändern.

APL/68000 besteht aus einem „Superset", einer Übermenge, des APL.SV. Letzteres ist derzeit eine Art „Industriestandard" (made by IBM). Damit ist das APL des ST bestens für Büro-Anwendungen gerüstet, wo meist Kompatibilität zu bestehender Software gefordert ist. Und APL selbst ist (nach Cobol und Fortran) der Industriestandard. Fragen Sie mal „Big Blue" IBM! – „Superset" heißt: da ist mehr drin. Stimmt. Es ist etwa möglich, auch Textstrukturen zu sortieren. Die Kriterien sind beliebig. Die Funktionen „Komprimieren" und „Expandieren" sind erweitert: APL/68000 läßt als linkes Argument auch Integer-Elemente zu. Für das maßstäbliche Vergrößern/Verkleinern von Bitmaps keine unwesentliche Erleichterung. Und neben dem 'Format'-Befehl und der System-Funktion 'QUAD-FMT' gibt es eine Cobol-ähnliche 'ALPHA'-Funktion, die für zusätzlichen Komfort bei der Anzeige von Tabellen sorgt. Kommentare können am Ende jeder Funktionszeile stehen. Mehrere unabhängige Ausdrücke in einer Zeile sind möglich.

Systemfunktionen

Für jedes APL gibt es Spracherweiterungen. Sie werden "Systemfunktionen" bzw. "Systemvariablen" genannt und beginnen mit einem '⎕'. Manche, wie '⎕IO', gibt es in jedem System (damit wird festgelegt, ob APL beim Zählen mit 0 oder mit 1 anfangen soll). Hier gibt es aber nur eine (höchst unvollständige) Übersicht bezogen auf Spezialitäten des APL/68000.

⎕A	Alphabet (Großbuchstaben)
⎕a	Alphabet (Kleinbuchstaben)
⎕BOX	– zur Umwandlung von Matrizen in Vektoren mit Delimitern, und umgekehrt (gültig für Zeichen und Zahlen!)
⎕CC	"Console Control" – Geräteunabhängige Bildschirmfunktionen
⎕D	"Digits" (Ziffern 0–9)
⎕DBR	"Delimited Blank Removal" – Entfernen überflüssiger Leerzeichen
⎕DR	– zur Kontrolle oder zum Ändern der (internen) Daten-Repräsentation
⎕FMT	– dient der Tabellenformatierung (mit Tabs, Texteinschüben, Rundung etc.)
⎕M	Liste der Monate
⎕OV	Overlay-Funktion. Damit lassen sich unterschiedliche Variablen (auch mit verschiedenen Strukturen und: Text und Zahlen gemischt!) sowie Funktionen in eine einzige Variable stecken bzw. wieder herausholen.
⎕SS	Funktion zum Suchen (wahlweise mit Ersetzen) von Zeichenketten in Vektoren. Es können mehrere Begriffe gleichzeitig gesucht/ersetzt werden.
⎕W	Liste der Wochentage

Wie gesagt, es handelt sich nur um eine kleine Auswahl. Es fehlen etwa die Funktionen zur Fehlerdiagnose, Trace- und Stop-Funktionen und viele mehr.

Abb. 2

Zugaben

APL ist eine „nackte" Sprache. Streng genommen gehören weder ein Dateisystem noch irgendwelche Interfaces zum Rechner zur Sprachdefinition – APL ist zwar eine Sprache, aber nicht unbedingt eine Computersprache. Die Güte einer Sprachversion hängt daher nicht unwesentlich von den gebotenen „Extras" ab. Dazu gehören neben einem Dateisystem alle Systemfunktionen und Systemvariablen. Beim ATARI ist zudem die Grafik-Unterstützung von Interesse – wo doch APL mit seinen Bitmaps und trigonometrischen Funktionen wie geschaffen ist für visuelle Anwendungen.

Aber der Reihe nach: Das Dateisystem des APL/68000 ist hervorragend. So einfach habe ich noch nie mit meinen Daten umspringen können (siehe Kasten). In Kombination mit den APL-Künsten bezüglich der Listenverarbeitung ist selbst bei komplexen Anwendungen ein separates Datenbankpaket nicht mehr nötig. Meine Datenbank heißt APL.

Grundsätzlich werden bei Systemfunktionen zwei unterschiedliche Wege beschritten. Die Einen, nicht schüchtern, packen rein, was immer gerade nützlich erscheint. Die Anderen sind eher vorsichtig, um größtmögliche Kompatibilität zu anderen Systemen zu wahren. MicroAPL zählt eher zur letzteren Sorte. Sicherlich ist alles enthalten, was ich von IBM-Rechnern her kenne. Die sparsamen Erweiterungen sind aber auf jeden Fall vom Feinsten: Keine Spielereien, sondern Hilfsmittel, bei denen sich mir nur noch die Frage stellt, warum „die Anderen" sie nicht haben. Am wichtigsten: Die Funktion zum Suchen und Ersetzen in Characterstrings und die Funktion zum Umwandeln von Matrizen in platzsparende Strings (mit Delimitern) und umgekehrt.

Die Funktionen zur Fehlerbehandlung sind sehr ausgefeilt. In meinen umfangreichen Anwenderprogrammen konnte ich auftretende Programmfehler nicht nur (unmerklich für den Benutzer) registrieren, sondern meist auch, dank detaillierter Information über Lokalisierung und Art des Fehlers, reparieren.

Für die Anbindung des APL an den ATARI gibt es eine Reihe von Bibliotheken (Workspaces), die im Lieferumfang enthalten sind. Alertboxen, Menüs und Dialogboxen sind mit Hilfe von Coverfunktionen sehr einfach zu handhaben. Stundenlange Sitzungen mit dem Resource-Construction Set entfallen. Auch die VDI-Funktionen stehen zur Verfügung. Hier sind meine Eindrücke etwas zwiespältig: Zum Erstellen von Geschäftsgrafiken, Diagrammen und Kurven sind die Coverfunktionen hervorragend geeignet, da sie sehr bequem zu handhaben sind (kein Stochern in den Kontrollstrukturen). Für CAD-Anwendungen ist aber der Overhead zu groß. Das ist ausgesprochen schade, gerade weil APL an sich für rechenintensive Applikationen wie geschaffen ist.

Nach Angaben von gdat wird es zwei Lösungen geben: MicroAPL arbeitet an einer Dokumentation der internen Schnittstelle zum VDI. Direkter Zugriff auf dieses Interface beschleunigt die Grafik auf VDI-Speed. Das konnte ich eigenhändig testen. Außerdem gibt es bei gdat einen Set von Funktionen, die direkt auf den Line-A Variablen operieren. Damit sind Geschwindigkeitsprobleme bei der Grafik pass. – Aber der Blitter spielt mit, wenn er soll.

Ein weiterer Workspace dient dem Zugriff auf das TOS-Dateisystem. Die darin enthaltenen Funktionen haben es mir gestattet, mit 1st Word editierte Texte einzulesen, um sie unter APL weiter zu „behandeln" und mit einer Adressdatei zu verknüpfen, die doch einiges mehr kann als 1st Mail. Auch DEGAS-Bilder (monochrom) konnten problemlos gelesen, verarbeitet und wieder abgelegt werden. Mit den APL-Sprachkünsten ist es eine reine Freude, Zeichnungen zu mischen, zu manipulieren (Drehen, Spiegeln ...), oder etwa Ausschnittsvergrößerungen herzustellen.

Das Dateisystem

Neben einem vollständigen Satz von Funktionen zur Bearbeitung von TOS-Dateien besitzt APL/68000 ein eigenes, höchst komfortables Dateisystem, welches völlig rechnerunabhängig arbeitet (und deshalb leicht portierbar ist). So geht's:

```
<Daten> ⍄ 123 1
```

'123' ist eine Dateinummer, die '1' die Datensatznummer. Die <Daten> sind beliebig. Das Zeichen '⍄' bedeutet: '**Datensatz schreiben**'. Wenn es die Datei '123' oder den Datensatz '1' noch nicht gibt – wen stört's? APL erledigt die Feinheiten intern. Das Lesen geht genauso einfach:

```
R ← ⍄ 123 1
```

Das wär's schon. Ein Öffnen oder Schließen der Datei ist überflüssig. Zusätzlich kann man ein (logisches) Laufwerk angeben – Das kann eine Floppy, ein Ordner oder ein beliebiges Subdirectory sein:

```
<Daten> ⍄[2] 123 2
```

Bis zu 10 logische Laufwerke können gleichzeitig angemeldet sein (und jederzeit programmgesteuert neu belegt werden). Für jedes Laufwerk richtet APL bei Bedarf eine TOS-Datei ein, die sich automatisch vergrößert, wenn es nötig ist. Darin können beliebig viele APL-Dateien mit beliebigen Datensätzen untergebracht sein.

Weitere Fähigkeiten in Stichworten:

- Einfügen von Datensätzen (<Daten> ⍄ 123 1.5)
- Infofunktionen (vorhandene Dateien/Datensätze; Größe von Files/Components; Datum des letzten Schreibzugriffs; ...)
- Löschen von Dateien oder Datensätzen
- Zugriffssperren (wichtig für Multiuser-Betrieb)
- Festlegung von Obergrenzen (Dateigrößen)

Es können hier nicht alle Features aufgezählt werden, aber auf jeden Fall ist dieses System wesentlich einfacher zu benutzen als zu beschreiben. Ausprobieren!

Abb. 3

Zukunftssicher

Manchen Entwickler überkommt der nackte Horror, wenn er an die Übertragung seiner Programme auf andere Rechner denkt. APL ist da recht angenehm: Es gibt Interpreter für alle gängigen Rechner, vom Micro bis zum Mainframe. Durch Coverfunktionen lassen sich maschinenabhängige Stellen

Benchmarks

Geschwindigkeitsvergleiche sind ein probates Mittel, die Leistungsfähigkeit einer Sprache zu beurteilen. Sie sind jedoch mit Vorsicht zu genießen. APL verlangt (im Gegensatz zu herkömmlichen Sprachen) Array-orientierte Algorithmen. Die Problematik wird im Vergleich mit dem GFA-Basic klar: Eine 1:1 - Umsetzung der Routinen aus der ST 7/86 in APL wäre zwar möglich, widerspricht aber diametral der in APL allgemein angewandten Vorgehensweise. Die APL-Zeiten spiegeln also eine sinngemäße Übertragung wider. Schleifen werden nicht verglichen, da sie in APL typischerweise der Kontrolle des Programmablaufs dienen, nicht aber dem "Durchleiern" von Strukturen. Das sind zwei Paar Schuhe ...

Nr.	Beschreibung	APL-Ausdruck	APL/68000	GFA-Basic
2	Inkrementierung von K=1 bis K=1000	K←⍳1000	0.01	0.37
3	Alle Grundrechenarten (1 bis 1000)	A←K÷K×K+K-K	1.50	1.02
4	Alle Grundrechenarten (A=K/2*3+4-5)	A←K÷2×3+4-5	1.41	1.01
7	wie 4, mit GOSUB..., geschachtelt, DIM	A←K÷2×3+4-5	1.41	2.89

- Die Benchmarks und die Zeiten für das Basic sind der ST-Computer 7/86, S. 87 entnommen. Die Rechengenauigkeit des Basic beträgt nur 11 Stellen, die des APL 15. Außerdem ist die DIM-Anweisung schon in jedem APL-Ausdruck implizit enthalten (siehe Benchmark 7). Alle Angaben sind in Sekunden. Die APL-Zeiten wurden in 100 Durchläufen ermittelt.

Die folgenden Benchmarks dienen dem Vergleich unterschiedlicher APL-Interpreter. Die Zeiten für das APL*PLUS/PC (STSC) und das PC-APL (IBM) sind der BYTE, März 1984, entnommen. Zwei Einschränkungen vorweg: Die in der BYTE getesteten APLs sind in der Zwischenzeit sicherlich verbessert worden (Tuning). Außerdem kann es ein PC (4.77 MHz) nicht mit dem ATARI-ST aufnehmen. Allerdings sind die Benchmarks mit dem Co-Prozessor 8087 gefahren worden. Ein Arithmetik-Chip ist im ST nicht eingebaut, obwohl er gerade APL sehr zugute käme. Daher ist sicher interessant, ob sich die Software-Arithmetik des APL/68000 im Vergleich mit der Hardware-Lösung behaupten kann. Alle Zeitangaben sind in Millisekunden und beziehen sich auf einen Durchlauf (gemittelt aus 100).

Nr.	Funktion	Syntax	APL/68000 (Atari-ST)	PC-APL (IBM-PC	APL*PLUS mit 8087)
1	Plus-Reduktion	Z←+/VI	6	90	102
2	Komprimieren	Z←∨/VL	1.6	0.4	3
3	Maximalwert ermitteln	Z←⌈/[1]MI	4.8	40	25
4	Potenzieren	Z←VI*.1	2640	390	2466*/282
5	Absolutwert bilden	Z←∣VR	51.4	80	79
6	Indizierung	Z←VR[VI[⍳20]]	10.8	20	14
7	Sortieren (absteigend)	Z←VI[⍒VI]	125.2	600	112
8	Entnehmen	Z←¯2 1↑MR	3.4	9	24
9	'Element von'	Z←VI∊VI	86.2	150	146
10	Transponieren	Z←2 1⍉MC	113.6	450	60
11	Äußeres Produkt, Characters	Z←VC∘.=VC	24.4	360	141
12	Äußeres Produkt, Integers	Z←(⍳50)∘.+⍳50	64.6	2530	439
13	Inneres Produkt, Reals	Z←VR⌊.+VR	124.8	210	341
14	Matrix-Division	Z←MR⌹10↑VR	207.6	70	1488
15	Fibonacci-Serie	(Listing 1)	1122.4	2200	3827
16	Multiplikation	Z←VR×3.14	155.6	100	136
17	Division	Z←VR÷3.14	269.6	110	142
18	Logarithmus	Z←⍟VR	1521	150	5094*/143
19	Sinus	Z←1○VR×.1	1759	411	12009*/438
20	Sieb des Eratosthenes	(Listing 2)	67.78 sec.	168 sec.	169.3 sec.

*) vergleichsweise die Zeit ohne eingebauten Arithmetik - Chip (Angabe aus BYTE, s.o.)

Die in Benchmarks 1 - 19 verwendeten Variablen sind wie folgt definiert:

```
MI←10 10⍴ VI←(500⍴0 1 0 0 1)/⍳500
VL←1 0 1 1 0 0 0 1 ◊ MR←10 10⍴ VR←VI+0.1 ◊ MC←26 26⍴ VC←⎕A
```

Abb. 4

```
  ∇FIB
  Z←1 1
L:→(100>⍴Z←Z,+/¯2↑Z)/L
```

Listing 1: Fibonacci-Serie

```
  ∇SIEVE
  I←1 ⋄ F←(8191⍴Y),E
Y:F[I+P×⍳⌊(8191-I)÷P←1+I+I]←N
N:→F[I←I+1]
E:(⍕Y+.=F),'PRIMES'
```

Listing 2: Sieb des Eratosthenes

Abb. 5

Bugs der V.6.00A

Es handelt sich um alle bis zum 30.7.87 gemeldeten Fehler. Sie sind sämtlich in der Version 6.05 behoben.

- Das Zeichen 'ß' ist von der Tastatur aus nicht ohne weiteres zugänglich.
- Die Tastaturumschaltung arbeitet nicht, wenn 'Caps Lock' gesetzt ist.
- Die Systemkommandos)LOAD,)SAVE und)COPY verursachen gelegentlich einen Systemabsturz.
- Zu Anfang einer Sitzung kann die (falsche) Meldung WS LOCKED erscheinen.
- ⎕CC 1 arbeitet falsch, wenn das logische APL-Window nicht an den Anfang gescrollt ist.
- Die Anwendung des SCAN-Operators kann in einem seltenen Fall zum Systemabsturz führen.
- Die Funktion DIALOG kann nur ein einzelnes editierbares Feld pro Dialogbox verarbeiten.
- Die Funktion STDIR zeigt keine Directory-Einträge an.
- Die Funktion ALERT führt keinen Syntax-Check aus.

Abb. 6

leicht isolieren. Wenn Sie mal Lust auf die SUN (Workstation unter UNIX) verspüren, sollten Sie die rechnerunabhängige Systemfunktion QUAD-CC benutzen: Selbst auf einem seriellen Terminal läuft Ihre Software ohne Änderungen. Interessanter ist allerdings die Portierung auf MAC oder AMIGA. Haben Sie schon mal eine Dialogbox für den ST entworfen, die sofort auf dem MAC läuft? Oder eine Geschäftsgrafik auf die AMIGA portiert? – Ich konnte mich davon überzeugen, daß beide Fälle möglich sind, ohne auch nur ein Zeichen im Code zu verändern.

Und die Zukunft auf dem ST? APL/68000 läuft nach Angaben von MicroAPL auf allen Modellen, vom alten 260-ST bis hin zum Mega-ST mit Blitter und 4 Megabytes RAM, sowie in allen Auflösungen. Da MicroAPL grundsätzlich keine unsauberen Tricks verwendet, hat kein ST-Besitzer Ärger mit neuen TOS-Versionen zu befürchten. Ein Test mit dem Mega-ST (4 Mb, neues TOS) verlief einwandfrei. Und da schon APL-Versionen für die CPU 68020 (mit Coprozessor-Unterstützung) existieren, ist auch für kommende ATARI-Entwicklungen vorgesorgt.

Der ST ist (dank GEM) leicht zu benutzen. Mit APL ist er nun auch leicht zu programmieren. Es soll Väter geben, die ihren schulpflichtigen Kindern einen ST nur unter der Bedingung hinstellen, mit APL zu arbeiten. Das könnte sich nämlich über eingesparte Nachhilfestunden schnell bezahlt machen. Für die Lösung von ad hoc-Problemen ist APL sowieso die Sprache der Wahl, und im Profi-Bereich hat APL seinen Platz im sogenannten „Prototyping" sowie bei Anwendungen, die flexibel und anpassungsfähig sein müssen. Und ein MUSS ist APL immer dort, wo es auf kurze Entwicklungszeiten ankommt. Ein Run-Time System ist kostenlos verfügbar. Damit ist die letzte Hürde für die Verbreitung professioneller Software gefallen.

Ich kann dem APL/68000-ST in der Version 6.05 bescheinigen, daß es alle Ansprüche erfüllt, die von einer professionellen Programmiersprache verlangt werden. Angesichts der vielen Features, die bei anderen Sprachen zu den Extras zählen (oder nicht existieren), ist der Preis von DM 428,– als „billig" zu bezeichnen. Es sollte mich wundern, wenn dieses APL nicht binnen kürzester Zeit vom Geheimtip zum Massenschlager avanciert. – Eigentlich schade um den Hauch von Exklusivität, den ich beim Gebrauch von APL auf dem ST bisher genießen konnte.

Wolfgang Thomas

Name:	**APL/68000-ST**
Hersteller:	**MicroAPL Ltd., London**
Preis:	**ca. DM 298,-**
Bezugsquelle:	**gdat mbH Stapelbrede 39 48 Bielefeld 1**

A-MAGIC

Atari ST und Amiga Hard- und Software-Produkte

Algorithmen und Datenstrukturen in PASCAL

Teil I

Im Laufe dieser Serie werde ich Sie, liebe Leser und ATARI-Freunde, nach und nach mit Begriffen der Programmierung, wie Queues, Listen, Bäume und einigen Sortierverfahren, bekanntmachen.

Als vermittelnde Computersprache habe ich PASCAL gewählt, weil PASCAL's klare, programmtechnische Formulierungen bei diesen doch recht abstrakten Problemen, den Kürzeln der Modesprache C eindeutig vorzuziehen sind. Dies soll jedoch die Freunde einer anderen Programmiersprache nicht abschrecken, denn prinzipiell ist der behandelte Stoff leicht auf jede Sprache umzusetzen, die des PRODEDURE/FUNCTIONS-Konzeptes mächtig ist, sowie über eine dynamische Speicherverwaltung verfügt (PASCAL, MODULA II, C, ...). Vorweg noch eine weitere technische Anmerkung. Ich habe den PASCALplus-Compiler (CCD) zur Übersetzung der Programmbeispiele benutzt. Dies spielt aber keine wesentliche Rolle, da im Laufe dieser Serie nur Standard PASCAL, gemäß Niklaus Wirth, benutzt werden soll. (An den wenigen Stellen, an denen ich vom Standard abweiche, werde ich es erwähnen.

Was ist überhaupt ein Stack?

Wie die deutsche Übersetzung des Begriffes 'Stack' schon sagt, handelt es sich um einen Stapel hier: um einen Stapel von Informationen. Um einen solchen Stapel zu verwalten, ist es notwendig, Operationen zum Aufbauen, Erweitern, Abbauen sowie der Kontrolle zu besitzen. Hierzu haben sich die folgenden Begriffe eingebürgert:

- CREATE, zum Erzeugen eines Stacks
- PUSH, zum Hinzufügen eines Stackelementes
- POP, zum Entfernen eines Stackelementes
- TOP, zum Auslesen des obersten Stackelementes und IS_EMPTY, zum Check auf den leeren Stack

Die Wirkungsweise dieser Operationen wird in Abb. 1a verdeutlicht.

PUSH(X,A)

POP(X)

Legende zu 1a

Stackpointer

Normales Stack-Element

Unterstes Stack-Element

Oberstes Stack-Element (Stackpointer)

Noch nicht eingefügtes Element

Abbildung 1a: Veranschaulichung der PUSH- und POP- Operationen

```
Eingabe Infix          ---> 3+7-2*(3+5-6*(3+1-(2*3+4))).
Modifizierte Postfix ---> 3#7#+2#3#5#+6#3#1#+2#3#*4#+-*-*-.
Auswertung             ---> -78
```

Abbildung 1b: Testlauf RECHNER.TOS

Gezeigt wird die Arbeitsweise der PUSH- und der POP-Operation. Der in den Stacks 'oben' abgebildete Bereich zeigt den Stackpointer, also den Zeiger, der auf das zuletzt 'abgestapelte' Element weist. Durch die PUSH-Operation (links) wird ein Element dem Stack hinzugefügt und der Stackpointer entsprechend umgesetzt. Die POP-Operation (rechts) entfernt das oberste Element aus dem Stack und setzt den Stackpointer auf das darunter liegende Element. Diese Art der Informationsspeicherung nennt man LIFO, für Last-in-first-out. Mit der in der Abbildung nicht erwähnten TOP-Operation erhält man die Informationen des obersten Stackelements, (auf das der Stackpointer hinweist), ohne dieses jedoch vom Stack abzutrennen. Ein weiterer Befehl ist die Stackkontrollfunktion IS_EMPTY. Da es nur mit den drei Operationen PUSH, POP und TOP möglich wäre, nicht vorhandene Stackelemente zu löschen, dient sie dazu, vor einem Zugriff mit POP oder TOP den Stack auf Elemente zu überprüfen. Betrachten Sie dazu das kurze Beispielprogramm.

Ausgehend von einem leeren Stack werden folgende drei Operationen durchgeführt:

```
PUSH(X,A);
POP(X);
POP(X);
```

Dies führt zwangsläufig zu einem Fehler, da auf dem Stack nur ein Element abgeladen wurde, zwei aber gelöscht wurden. Deshalb ist es in der Anwendung von Stacks unbedingt ratsam, vor jeder TOP- oder POP-Operation zu überprüfen , ob der betrachtete Stack nicht leer ist.

```
IF NOT IS_EMPTY(X) THEN
POP(X);
```

Programmierung von Stacks

Kommen wir nun zur Programmierung von Stacks. Wenn Sie unsere bisherigen Definitionen rekapitulieren, werden Sie feststellen, daß der Stack durch wiederholtes Anwenden von PUSH praktisch beliebig (unendlich) groß werden kann. Dies nennt man eine unendliche Datenstruktur. Wenn sich also die maximale Stackgröße nicht abschätzen läßt, ist es erforderlich, auf die dynamische Speicherverwaltung zurückzugreifen. (Spätestens hier kommen unsere BASIC-Freunde ins Schwitzen.)

Im Listing der Datentypen (Listing 1a) sehen Sie, daß ich hierzu den Pointer STACK definiert habe. Die abzuspeichernden Informationen (STACK_DATA) habe ich als INTEGER gewählt, hier kann aber jeder beliebige Typ stehen. Ein STACK_ELEMENT besteht somit aus STACK_DATA und einem Zeiger auf das darunterliegende Stackelement (NEXT). Mit dieser Struktur ist es möglich, den Stack für die geforderten, unendlichen Merkmale bis zur Arbeitsspeichergröße des ATARI ST aufzublasen. Die Arbeitsweise sieht folgendermaßen aus:

1. Vereinbarung einer Variable vom TYP STACK: VAR X : STACK; Machen Sie sich an dieser Stelle bitte klar, daß es sich bei X, aufgrund der Zeigerstruktur unseres Datentypes, eigentlich nicht um einen Stack, sondern lediglich um einen Stackpointer handelt. Trotzdem werde ich im Folgenden immer vom 'Stack X' sprechen.
2. Initialisierung dieser Variablen X mit CREATE zu NIL, um den leeren Stack zu kennzeichnen.
3. Durch jeden Aufruf von PUSH wird dem Stack X, zunächst dem leeren Stack, ein Element A hinzugefügt, indem mittels NEW ein neues Element (NEW_ELEMENT) erzeugt wird. Die Daten dieses neuen Elementes werden zu A initialisiert. Der darunterliegende Stack ist ja unser alter Stack, deshalb ist er auch Nachfolger des neuen Elementes. Schließlich wird die Stackspitze (Stackpointer) X auf das neue Element gerichtet.
4. Mit TOP wird, im Falle des nicht leeren Stacks, das oberste Element des Stacks über den Funktionsreturn zurückgeliefert. Im Falle des leeren Stacks ist der Rückgabewert undefiniert.
5. Mit POP kann solange das oberste Element des Stacks gelöscht werden wie noch Elemente vorhanden sind. Am Schluß muß die Rückgabe eines jeden Elementes an das Betriebssystem mittels DISPOSE, erfolgen. Werden nicht vorhandene Stackelemente gelöscht, so ruft dies keinen Fehler hervor, sondern erzielt lediglich keine Wirkung.
6. Mit IS_EMPTY kann modulexplizit und wird modulimplizit der Stack X auf Elemente überprüft.

Die Funktionalität der Funktionen und Prozeduren, soweit Sie sie nicht schon aus den Programmbeispielen entnommen haben, findet sich leicht im Listing. Ein X in der Parameterliste steht hier für einen Stack, ein A für die Daten eines Stacks.

Wenn Sie unsere bisherigen Vereinbarungen nocheinmal betrachten, werden Sie feststellen, daß an einer Stelle ein Widerspruch besteht. Zunächst habe ich betont, daß ein Stack beliebig groß werden kann, dann habe ich gesagt, daß ein Stack sich bis zur Arbeitsspeichergröße ausdehnen darf, was natürlich eine unumgängliche Einschränkung darstellt. Natürlich wäre es gut, auch bei der PUSH-Operation, eine Sicherheitsabfrage zu besitzen, um nachzuprüfen, ob der benötigte Speicherplatz für STACK_DATA noch vorhanden ist. Leider bietet Standard PASCAL hierzu keinerlei Möglichkeit. Als Anregung für das Selbstprogrammieren sei erwähnt, daß PASCAL-plus zur Abschätzung des freien Speichers die parameterlose Funktion MEMAVAIL besitzt und sich der benö-

```
  Dieses Modul stellt die Typen fuer den abstrakten Datentyp STACK zur
  Verfuegung.

  Reservierte Woerter : STACK, STACK_ELEMENT, STACK_DATA, DATA und NEXT.

  programmiert        : April 87

  von                 : Dirk Brockhaus
)

STACK = ^STACK_ELEMENT;

STACK_DATA = INTEGER;

STACK_ELEMENT = RECORD
                 DATA : STACK_DATA;
                 NEXT : STACK;
                END;
```

Listing 1a

tigte Speicherplatz mit der Funktion SIZEOF(STACK_DATA) berechnen läßt.

Was soll das Ganze?

Wenn Sie nicht schon ein wenig in die Materie der Informatik eingedrungen sind, werden Sie sich irgendwann beim Lesen dieses Artikels gefragt haben, 'Was soll das Ganze?' oder, 'Wozu sind Stacks eigentlich gut ?'. Antwort: Stacks sind praktisch unentbehrlich. Ein Compiler kommt nicht ohne sie aus, das Betriebssystem legt seine Unterprogrammrücksprungadressen in Stacks ab (Supervisorstack) und nicht zuletzt jedes Anwendungsprogramm macht regen Gebrauch von Stacks (Userstack, Rekursion, ...). Doch das sind alles Stacks, die den 'normalen' Anwendungsprogrammierer wenig interessieren, abgesehen natürlich von ihren höchst positiven Auswirkungen.

Ein allgemeineres Beispiel für die Anwendungsgebiete von Stacks ist die Analyse und Auswertung arithmetischer Formeln. Wenn es darum geht, in einem Programm eben eine Zeile mit einer mathematischen Formel zu ändern, werden zumeist die Interpreterfreunde beneidet, denn diese hier so beiläufig erledigte Arbeit stellt Compilersprachen vor einige Probleme, da eine Änderung des Quellcodes als Lösung ausscheidet.

Anwendung von Stacks: Formelauswertung

Als Demonstration der Möglichkeiten von Stacks stelle ich Ihnen nun ein vereinfachtes Modell zur Auswertung arithmetischer Ausdrücke vor. Grundsätzlich ist es möglich, arithmetische Ausdrücke in dreierlei Form darzustellen.

1. Die Notation mit zwei Operanden und dazwischen stehendem Operator: a+b Diese Art der Schreibweise wird infix-Notation genannt.
2. Die präfix-Notation mit vorstehendem Operator und nachstehenden Operanden: +ab
3. Die uns interessierende postfix-Notation mit nachstehendem Operator und vorstehenden Operanden: ab+, die hier von Interesse ist.

Ziel des Programmes wird es sein, einen arithmetischen Ausdruck während der Eingabe von der dem Menschen angepaßten infix-Notation in die computerfreundliche postfix-Notation zu bringen. Die Computerfreundlichkeit der postfix-Notation liegt daran, daß sämtliche Prioritätsregeln alleine durch die Abfolge von Operanden und Operatoren berücksichtigt werden. Betrachten Sie hierzu folgendes Beispiel:

infix 1★(2+3) -> postfix 1 2 3 + ★

Die Auswertung dieser postfix-Schreibweise erfolgt von links nach rechts. Hierzu werden zunächst die Operatoren betrachtet. Man stößt auf den '+'-Operator. Dieser bezieht sich auf die beiden direkt links neben ihm stehenden Operanden, also '2' und '3'. Das Ergebnis sowie die erste Zahl bilden nun ihrerseits wieder die Operanden für den nachfolgenden '★'-Operator. Wie Sie sehen, ist die Klammerung und Punkt-vor-Strich-Rechnung hierbei bereits implizit in der Abfolge der Operatoren berücksichtigt. Wie sich in der untenstehenden algorithmischen Betrachtung des Problemes herausstellen wird, werden sowohl bei der Umwandlung in die postfix-Form als auch bei der Auswertung dieses Ausdrucks Stacks benötigt.

Zur Eingabe zugelassen werden arithmetische Ausdrücke in infix-Notation, die ganze Zahlen enthalten, sowie die Zeichen ('+','-','★','(',')'), also drei der Grundrechenarten plus Klammerausdrücken. Um dem Programm das Ende eines Ausdruckes mitteilen zu können, führe ich noch das Endesymbol '.' ein. Wenn Sie nun das einfache Beispiel '11-1' betrachten, so stellen Sie fest, daß die postfix-Notation, '111-', nicht mehr eindeutig ist, da die Trennlinie zwischen den beiden Zahlen nicht mehr bestimmt werden kann. Deshalb ist es von Seiten des Programmes erforderlich, ein Zeichen in die postfix-Form einzufügen, das das Ende einer ganzen Zahl symbolisiert. Hierzu habe ich '#' gewählt. In unserem einfachen Beispiel sieht die postfix-Form nun wie folgt aus: '11#1#-'. Daraus ergibt sich jetzt ('0'-'9','+','-','★','(',')','.','#') als Gesamtalphabet, wobei '#' von der Eingabe ausgeschlossen ist. Weiterhin läßt sich feststellen, daß die Zahlen in allen Notationen (infix, postfix und präfix) die gleiche Abfolge besitzen. Dies bedeutet, daß Zahlen sofort bei der Eingabe in die postfix-Notation übernommen werden können. Lediglich die Folge der Operatoren ist der postfix-Form anzupassen, wobei die Prioritäten der Punkt-vor-Strich-Rechnung und der Klammern berücksichtigt werden müssen. Zur Berücksichtigung dieser Prioritäten wird nun ein Stack benutzt.

1. Eine geöffnete Klammer wird 'abgestackt'.
2. Bei einer geschlossenen Klammer werden alle Operatoren vom Stack geholt und in die postfix-Form übernommen, bis eine offene Klammer erscheint. Diese wird überschrieben.
3. Bei einem Operator werden alle auf dem Stack liegenden Operatorzeichen, die mindestens gleiche Priorität wie das Gelesene haben ausgegeben. Erscheint ein Zeichen mit kleinerer Priorität auf dem Stack, so wird das aktuelle Zeichen abgestackt. '(' hat hierbei die niedrigste Priorität.

Programm

Mit diesen Vorbetrachtungen können wir uns nun dem Programm zuwenden (Listing 1c). Als Stack habe ich den in Listing 1a) und Listing 1b) beschriebenen Integer-Stack benutzt.

VAR X: STACK;

Zum Speichern der postfix-Form, zwischen EINGABE und AUSWERTUNG benutze ich einen ARRAY:

AUS : ARRAY [1..80] OF CHAR;

Je nach gewünschter Größe der Ausdrücke müßte die Grenze 80, nach unten oder nach oben verschoben werden.

Eingabe

Kommen wir nun zur PROCEDURE EINGABE. Sie besitzt drei Unterroutinen. Die erste Unterroutine SCHREIBE, schreibt einen Charakter in den Zwischenspeicher für die postfix-Form (AUS). Die nächsten beiden Unterroutinen übernehmen Aufgaben aus dem oben beschriebenen Algorithmus. CLEAR arbeitet den Stack bis auf die nächste geöffnete Klammer ab (Punkt 2). PRIOR weißt jedem Operator seine Priorität zu. In der PROCEDUR selber findet sich nun der Algorithmus wieder. Nach einigen Einleitungspreliminarien, Erzeugung des Stack und Abstacken einer geöffneten Klammer zur Stackbödenmakierung, beginnt die WHILE-Schleife, die die Eingabe bis zur Endemarkierung einließt. Hier wird zunächst ein Zeichen gelesen. Handelt es sich um eine Zahl, wird diese direkt in den AUS-ARRAY geschrieben. Bei einem anderen Zeichen wird weiter differenziert. Zunächst muß hier überprüft werden, ob der Vorgänger eine Zahl war, weil in diesem Falle noch die Zahlendemarkierung '#' nach AUS zu schreiben ist. Die Analyse des Zeichens selber (CASE EIN OF) erfolgt in völliger Analogie zum Algorithmus (siehe Punkt 1–3).

Ausgabe

Die einzige Funktion der darauffolgenden PROCEDURE AUSGABE, besteht darin, den ARRAY AUS auszugeben.

```
   Dieses Modul stellt die Operationen auf dem abstrakten Datentyp STACK
   zur Verfuegung:

   Funktionalitaet der Operationen:

    - CREATE: () --> STACK
    - IS_EMPTY: STACK --> BOOLEAN
    - PUSH: STACK X STACK_DATA --> STACK
    - POP: STACK --> STACK
    - TOP: STACK --> STACK_DATA

   Reservierte Woerter : CREATE, IS_EMPTY, PUSH, POP und TOP

   programmiert        : April 87

   von                 : Dirk Brockhaus
 )

PROCEDURE CREATE(VAR X:STACK);
 BEGIN
  X:-NIL;
 END;

FUNCTION IS_EMPTY(X:STACK):BOOLEAN;
 BEGIN
  IS_EMPTY:-X-NIL;
 END;

PROCEDURE PUSH(VAR X:STACK; A:STACK_DATA);
 VAR NEW_ELEMENT : STACK;
  BEGIN
   NEW(NEW_ELEMENT);
   NEW_ELEMENT^.DATA:-A;
   NEW_ELEMENT^.NEXT:-X;
   X:-NEW_ELEMENT;
  END;

 PROCEDURE POP(VAR X : STACK);
  VAR OLD_ELEMENT : STACK;
  BEGIN
   IF NOT IS_EMPTY(X) THEN
    BEGIN
     OLD_ELEMENT:-X;
      X:-X^.NEXT;
      DISPOSE(OLD_ELEMENT);
    END;
  END;

 FUNCTION TOP(X : STACK):STACK_DATA;
  BEGIN
   IF NOT IS_EMPTY(X) THEN
    TOP:-X^.DATA;
  END;
```

Listing 1b

Auswertung

Hierauf folgt der zweite Hauptteil des Programmes, die AUSWERTUNG. Auch hier sind Theorie und Praxis nicht besonders unterschiedlich. Eine WHILE-Schleife über dem ARRAY AUS wird solange besucht, bis das Wortendesymbol '.' erscheint. Ist der untersuchte Charakter eine Ziffer, so werden die nachfolgenden Charakter bis zur Zahlendemarkierung '#' untersucht und in einen INTEGER-Wert umgesetzt (WERT). Dieser wird zwecks weiterer Betrachtung gePUSHt. Handelt es sich jedoch um einen Operator, so werden die zwei obersten Operanden vom Stack geholt und mittels dieses Operators verknüpft. Hierbei ist auf die Reihenfolge der Operatoren zu achten, den sie stehen ja umgekehrt im Stack. Das Ergebnis wird dann wieder gePUSHt. Ist die postfix-form ganz abgearbeitet(Ende WHILE), so liegt das Ergebnis oben auf dem Stack TOP(X).

Main

Das Hauptprogramm beschränkt sich schließlich auf den Aufruf der drei oben beschriebenen Funktionen. Abbildung 1b) zeigt einen Probedurchlauf. Zunächst erfolgt die Eingabe der Formel in infix. Als Ergebnis folgt dieselbe Formel in postfix sowie das errechnete Resultat.

Vorausschau

Hiermit wäre ich mit den Stacks am Ende. Der Ausbau des obigen, kleinen Demoprogrammes oder andere Stackanwendungen seien dem interessierten Leser überlassen. In der nächsten Folge von Algorithmen & Datenstrukturen werde ich Queues (Warteschlangen) vorstellen. Hier werden auch wieder zwei kleine Module zur eigenen Implementierung von Queues sowie ein Beispielprogramm zu erwarten sein. (Sig)

Dirk Brockhaus

```
Dieses Programm dient zur Demonstration der Faehigkeiten des Types STACK.
Hierzu wird ein arithmetischer Ausdruck, der ganze Zahlen, +,- und *
enthalten darf, zunaechst, mittels eines Stack, in die Postfix- Form
ueberfuehrt und dann, ebenfalls mit einem Stack, ausgewertet.

programmiert : April 87

von          : Dirk Brockhaus

Anmerkungen  : - Die beiden Kommentare mit $ beginnend sind eine
                 Abweichung vom Standard und veranlassen den eingebauten
                 Praeprozessor von PASCAL+ den folgenden Filenamen
                 einzufuegen.

               - Das vorliegende Problem erfordert eigentlich zwei
                 STACK- Typen, naemlich ein INTEGER- STACK und einen
                 CHAR- STACK.
                 Der Einfachheit halber, wird deshalb in EINGABE ein
                 CHAR- STACK simuliert, indem nicht die Buchstaben,
                 sondern ihre Ordnungszahlen gepusht werden.

               - Dieses Programm dient nur der Demonstration.
                 Um den Rahmen dieser Publikation nicht zu sprengen, wird
                 deshalb keinen Check der Eingabe durchgefuehrt.
                 Bei falschen Ausdruecken koennten die Ergebnisse
                 ueberraschen ...
}

PROGRAM RECHNER (INPUT,OUTPUT);

TYPE {$I STACKTYP.PAS}

VAR X        : STACK;
    AUS      : ARRAY [1..80] OF CHAR;
    AUS_PTR  : INTEGER;

    {$I STACK.PAS}

PROCEDURE EINGABE;

  VAR EIN,
      VORGAENGER : CHAR;

  PROCEDURE SCHREIBE(CH : CHAR);

    BEGIN
      AUS_PTR:=AUS_PTR+1;
      AUS[AUS_PTR]:=CH;
    END;

  PROCEDURE CLEAR;

    BEGIN
      WHILE CHR(TOP(X))<>'(' DO
        BEGIN
          SCHREIBE(CHR(TOP(X)));
          POP(X);
        END;
    END;

  FUNCTION PRIOR (CH : CHAR) : INTEGER;

   BEGIN
     CASE CH OF
       '('       : PRIOR:=0;
       '+','-'   : PRIOR:=1;
       '*'       : PRIOR:=2;
     END;
   END;

  BEGIN
   AUS_PTR:=0;
   CREATE(X);
   EIN:='(';
   PUSH(X,ORD(EIN));
   WHILE EIN<>'.' DO
     BEGIN
       READ(EIN);
       IF EIN IN ['0'..'9'] THEN
         SCHREIBE(EIN)
       ELSE
         BEGIN
           IF VORGAENGER IN ['0'..'9'] THEN
             SCHREIBE('#');
           CASE EIN OF
             '('           : PUSH(X,ORD(EIN));
             ')'           : BEGIN
                               CLEAR;
                               POP(X);
                             END;
             '+','-','*'   : BEGIN
                               WHILE PRIOR(CHR(TOP(X)))>=PRIOR(EIN) DO
                                 BEGIN
                                   SCHREIBE(CHR(TOP(X)));
                                   POP(X);
                                 END;
                               PUSH(X,ORD(EIN));
                             END;
```

(Fortsetzung auf S. 82)

(Fortsetzung von S. 80)

```
          END;
         END;
        VORGAENGER:=EIN;
      END;
      CLEAR;
      POP(X);
      SCHREIBE('.');
    END;

  PROCEDURE AUSGABE;

    VAR I : INTEGER;

    BEGIN
      FOR I:=1 TO AUS_PTR DO
        WRITE(AUS[I]);
      WRITELN;
    END;

  FUNCTION AUSWERTUNG : INTEGER;

    VAR WERT  ,
        ZAHL1 ,
        ZAHL2 ,
        I     : INTEGER;

    BEGIN
      CREATE(X);
      I:=1;
      WHILE AUS[I]<>'.' DO
        BEGIN
          IF AUS[I] IN ['0'..'9'] THEN
            BEGIN
              WERT:=ORD(AUS[I])-ORD('0');
              I:=I+1;
              WHILE AUS[I]<>'•' DO
                BEGIN
                  WERT:=WERT*10+ORD(AUS[I])-ORD('0');
                  I:=I+1;
                END;
              PUSH(X,WERT);
            END
          ELSE
            BEGIN
              ZAHL2:=TOP(X);
              POP(X);
              ZAHL1:=TOP(X);
              POP(X);
              CASE AUS[I] OF
                '+' : PUSH(X,ZAHL1+ZAHL2);
                '-' : PUSH(X,ZAHL1-ZAHL2);
                '*' : PUSH(X,ZAHL1*ZAHL2);
              END;
            END;
          I:=I+1;
        END;
      AUSWERTUNG:=TOP(X);
      POP(X);
    END;

BEGIN
  WRITE('Eingabe Infix             ---> ');
  EINGABE;
  WRITELN;
  WRITE('Modifizierte Postfix ---> ');
  AUSGABE;
  WRITELN('Auswertung              ---> ',AUSWERTUNG);
(THE) END.
```

Listing 1c

Einkaufsführer

Hier finden Sie Ihren Atari Fachhändler

Anzeigenschluß Heft 12/87: 20. Oktober 1987

1000 Berlin

alpha computers g.m.b.h.
u. a. alphatronic, atari, commodore, dai, epson, sord mit pips, nec
hard-/software nach maß — servicetechnik
Kurfürstendamm 121a, 1000 Berlin 31 (Halensee)
Telefon 030/891 10 82

Computare
Keithstr. 18-20 • 1000 Berlin 30
030/21 390 21
186 346 com d

DATAPLAY
Bundesallee 25 · 1000 Berlin 31
Telefon: 030/861 91 61

ATARI
... wir machen Spitzentechnologie preiswert.
DIGITAL COMPUTER
Verkaufsbüro (1.OG) 1 Berlin 12
Knesebeckstr. 78 · Tel. 882 77 91
Software Hardware Beratung Zubehör Service Literatur

RUNOW
Büroelektronik
Keithstraße 26 · 1000 Berlin 30
☎ 26 111 26

1000 Berlin

Steglitz Schloßstraße
030/79001-418
Ihre Tür zur Zukunft:
karstadt-computer-center
hardware · software · problemlösungen

Ihre Tür zur Zukunft:
KARSTADT computer-center
hardware · software · problemlösungen
Berlin, Hermannplatz, Telefon (0 30) 6 90 81

Computershop Edith Behrendt
Fürbringerstraße 26 · 1000 Berlin 61
Telefon 0 30/6 91 76 66

ATARI
... wir machen Spitzentechnologie preiswert.
Vertragshändler
UNION ZEISS
Kurfürstendamm 57 • 1000 Berlin 15
Telefon 32 30 61

2000 Hamburg

Bit Computer Shop
Osterstraße 173 · 2000 Hamburg 20
Telefon: 040/49 44 00

Createam
Computer Hard & Software
Bramfelder Chaussee 300 · 2000 Hamburg 71
Telefon Sa. Nr. 0 40/641 50 91

2000 Hamburg

Gerhard u. Bernd Waller GbR
Computer & Zubehör-Shop
Kieler Straße 623
2000 Hamburg 54
☎ 040/570 60 07 + 570 52 75

Hardware
Software
Beratung
Service
HCAG HABA COMPUTER AG
ATARI Systemfachhändler
Münsterstraße 9 · 2000 Hamburg 54
Telefon 040/56 60 1-1

GMA mbH
040/2 51 24 16/17
Systemhändler
Wandsbeker Chaussee 58
2000 Hamburg 76

2000 Norderstedt

sellhorn
Ulzburger Str. 2
2000 Norderstedt
Tel. 0 40/527 30 47

2120 Lüneburg

Sienknecht
Bürokommunikation
Beratung - Verkauf - Werkstatt
Heiligengeiststr. 20, 2120 Lüneburg
Tel. 0 41 31 / 4 61 22, Btx 40 24 22
Mo.-Fr. 9⁰⁰-18⁰⁰ und Sa. 9⁰⁰-13⁰⁰

2210 Itzehoe

Der Computerladen
Innaber Ulrich Bübel Martin Kopplow
Coriansberg 2 · 2210 Itzehoe
Telefon (0 48 21) 33 90/91

2300 Kiel

MCC
MicroComputerChrist
Die Welt der Computer
Dreiecksplatz Nr. 7
2300 Kiel 1 · ☎ 04 31/56 70 42

2350 Neumünster

SM SOFTWARE + COMPUTER GMBH
Klosterstraße 2 · 2350 Neumünster
Telefon (0 43 21) 4 39 33

2390 Flensburg

ecl electronic computer laden ohg
Norderstr. 94-96 · D-2390 Flensburg
(0461) 2 81 81 & 2 81 93

2800 Bremen

PS-DATA
Doventorsteinweg 41
2800 Bremen
Telefon 04 21 - 17 05 77

2850 Bremerhaven

HEIM- UND PERSONALCOMPUTER
Kurt Neumann
Georgstraße 71
2850 Bremerhaven
Tel. 04 71/30 21 29
HARDWARE · SOFTWARE · PAPIERWARE

2940 Wilhelmshaven

Radio Tiemann
ATARI-Systemfachhändler
Markstr. 52
2940 Wilhelmshaven
Telefon 0 44 21 - 2 61 45

2950 Leer

S&F DATENTECHNIK
- HARDWARE-SOFTWARE
- SYSTEM-ENTWICKLUNG
- ORGANISATION
- EDV-SCHULUNG
- EDV-BERATUNG
- SERVICE-WARTUNG

Augustenstraße 3 · 2950 Leer
Telefon 04 91 - 45 89

3000 Hannover

• Software • Hardware • Organisation • Beratung • Schulung
ProComputer Hannover PCH
Großer Hillen 6 · 3000 Hannover 71
Telefon (0511) 52 25 79

DATALOGIC COMPUTERSYSTEME
ATARI ST · BERATUNG
COMPUTER · SERVICE
HARDWARE · VERKAUF
SOFTWARE
CALENBERGER STR. 26
3000 HANNOVER 1
TEL. 0511 · 32 64 89

COM DATA
Am Schiffgraben 19 · 3000 Hannover 1
Telefon 05 11 - 32 67 36

trendDATA Computer
IBM · EPSON · TRIUMPH ADLER
HEWLETT PACKARD · ATARI etc.
trendDATA Computer GmbH
Am Marstall 18-22 · 3000 Hannover 1
Telefon (05 11) 1 66 05-0

3040 Soltau

F & T Computervertrieb
Am Hornberg 1
(Industriegeb. Almhöhe)
3040 Soltau
Tel. 0 51 91/1 65 22

3100 Celle

Ludwig Haupt jr.
Büro-Einkaufs-Zentrum
Gerhard-Kamm-Straße 2
Ruf 8 30 45, Postfach 140
3100 Celle

3150 Peine

Wieckenberg & Schrage GmbH
Computertechnik
Hard- u. Software
Woltorfer Str. 8, 3150 Peine
Tel. 0 51 71/60 52/3 o. 0 51 73/79 09

3170 Gifhorn

COMPUTER-HAUS GIFHORN
Braunschweigerstr. 50
3170 Gifhorn
Telefon 0 53 71 - 5 44 98

3300 Braunschweig

COMPUTER STUDIO BRAUNSCHWEIG
Rebenring 49-50
3300 Braunschweig
Tel. (05 31) 33 32 77/78

3400 Göttingen

Büroeinrichtungs-Zentrum
Wiederholdt
3400 Göttingen-Weende
Wagenstieg 14 – Tel. 05 51/38 57-0

3500 Kassel

Hermann Fischer GmbH
autorisierter ATARI-Fachhändler
Rudolf-Schwander-Str. 5-13
3500 Kassel
Tel. (05 61) 70 00 00

RUDOLPH
Computersysteme & Bürotechnik
Einzel+Großhandel Im+Export
Telefon: 0561/472737
Telex Fax Btx 0561472739
Computerstudio: 3500 Kassel
Frankfurterstraße 311
Versand: 3501 Hoof Pf. 1160

3550 Marburg

L W M
COMPUTER SERVICE
Bahnhofstraße 26b
3550 Marburg/Lahn
✆ 0 64 21 - 6 22 36

4800 Bielefeld

MICROTEC
Software
Hardware
Beratung
Service

Carl-Severing-Str. 190
4800 Bielefeld 14
Telefon: 05 21/45 99-150
Telex : 9 37 340 krab d
Telefax: 05 21/45 99-123

5000 Köln

BÜROMASCHINEN
braun
AM RUDOLFPLATZ GmbH
5000 KÖLN 1
RICHARD-WAGNER-STR. 39
RUF: 02 21/21 91 71

5010 Bergheim

Computerstudio
HÖLSCHER
EDV-Beratung · Organisation
Programmierung · Home/Personal-Computer
Software · Zubehör · Fachliteratur
Zeppelinstr. 7 · 5010 Bergheim
Telefon 0 22 71 - 6 20 96

5060 Bergisch-Gladbach

Computer Center
Buchholzstraße 1
5060 Bergisch-Gladbach
Telefon 0 22 02 - 3 50 53

5090 Leverkusen

Rolf Rocke
Computer-Fachgeschäft
Auestraße 1
5090 Leverkusen 3
Telefon 0 21 71/26 24

5200 Siegburg

Computer Center
Luisenstraße 26
5200 Siegburg
Telefon 0 22 41/6 68 54

5400 Koblenz

SCHMITT COMPUTERSYSTEME
Casinostraße 40
5400 Koblenz
✆ 02 61 - 3 65 28

5457 Straßenhaus

DR. AUMANN GMBH
Computer-Systeme
Schulstr. 12
5457 Straßenhaus
Telefon 0 26 34 - 40 81/2

5500 Trier

bürocenter
LEHR
Güterstr. 82 - 5500 Trier
☎ 06 51 - 2 50 44
Fordern Sie unsere Zubehör-Liste an.

5540 Prüm

ATC COMPUTER
J. M. ZABELL
Ritzstraße 13 · Pf. 10 51
5540 PRÜM
– Tel.: 0 65 51 - 30 39 –

5600 Wuppertal

Jung am Wall
Wall 31—33
5600 Wuppertal 1
Telefon 02 02/45 03 30

MEGABYTE
Computer Vertriebs GmbH
Friedrich-Engels-Allee 162
5600 Wuppertal 2 (Barmen)
Telefon (02 02) 8 19 17

5630 Remscheid

COM SOFT
Scheiderstr. 12 · 5630 Remscheid
Telefon (0 21 91) 2 10 33 - 34

5800 Hagen

ATARI
wir machen Spitzentechnologie preiswert.
Vertragshändler Axel Böckem
Computer + Textsysteme
Eilper Str. 60 (Eilpezentrum) · 5800 Hagen
Tel. 0 23 31/7 34 90

5900 Siegen

HeesComputer
Vertriebs III GmbH
Hardware · Software · Schulung
Siegen · Weidenauer Str. 72 · ☎ 02 71/7 34 95

6000 Frankfurt

Müller & Nemecek
Kaiserstraße 44
6000 Frankfurt/M.
Tel. 0 69 - 23 25 44

GES-COMPUTER
GESELLSCHAFT FÜR EDV UND SOFTWARE mbH

Filiale Frankfurt
Hartmann-Ibach-Str. 63
6000 Frankfurt 60
Tel.: (0 69) 46 20 41

Filiale Hanau
Steinheimer Str. 22
6450 Hanau
Tel.: (0 61 81) 2 48 26

WAIZENEGGER
Büroeinrichtungen
Kaiserstraße 41
6000 Frankfurt/M.
☎ 0 69/2 73 06-0

ATARI
... wir machen Spitzentechnologie preiswert.
Vertragshändler
jetzt bei uns.
bco
Büro-Computer + Organisations GmbH
Oederweg 7—9
6000 Frankfurt/M. 1
☎ (0 69) 55 04 56-57
Hardware * Software * Beratung * Service

SCHMITT COMPUTERSYSTEME
Hanauer Landstr. 417
6000 Frankfurt/M. 1
Tel: 0 69/41 02 71
FAX: 0 69/42 57 71

6100 Darmstadt

Heim
Büro- und Computermarkt
Heidelberger Landstraße 194
6100 Darmstadt-Eberstadt
✆ 0 61 51/5 60 57

ATARI Systemfachhändler
PROFI COMPUTER STUDIO
KARSTADT Aktiengesellschaft
Elisabethenstr. 15 · 6100 Darmstadt
Luisencenter · Tel. 0 61 51 - 10 94 20

6100 Darmstadt

SCHMITT COMPUTERSYSTEME
Mühlstraße 76
6100 Darmstadt
✆ 06151-245 74

6200 Wiesbaden

SCHMITT COMPUTERSYSTEME
Rheinstraße 41
6200 Wiesbaden
✆ 06121-307330

6240 Königstein

KFC COMPUTERSYSTEME
Wiesenstraße 18
6240 Königstein
Tel. 06174-3033
Mail-Box 06174-5355

6300 Gießen

Ihre Tür zur Zukunft:
KARSTADT
computer-center
hardware · software · problemlösungen
Gießen, Seltersweg 64, Telefon (0641) 7004-318

Schneider ATARI Commodore
BAUMS
BÜRO · ORGANISATION
Bahnhofstr. 26 · 6300 Gießen
Telefon: 0641/71096

INTERFACE
Schnittstelle zwischen Mensch und Atari ST Computer
Thomas Heß
Asterweg 10 · 6300 Gießen
Telefon 0641/39153

6330 Wetzlar

Fachmarkt für Computer u. Unterhaltungs-electronic in Wetzlar,
Einkaufszentrum Bahnhofstraße, Tel. (06441) 48566

6400 Fulda

Schneider ATARI Commodore
WEINRICH
BÜRO · ORGANISATION
Ronsbachstraße 32 · 6400 Fulda
Telefon: 0661/492-0

6457 Maintal

Landolt-Computer
Beratung · Service · Verkauf · Leasing
Wingertstr. 112
6457 Maintal/Dörnigheim
Telefon 06181-45293

6500 Mainz

ELPHOTEC
Computer Systeme
Ihr Atari Systemhändler
mit eigenem Service-Center
Walpodenstraße 10
6500 Mainz
Telefon 06131-231947

SCHMITT COMPUTERSYSTEME
Karmeliterplatz 4
6500 Mainz
✆ 06131-234223

6520 Worms

ORION
Computersysteme GmbH
Friedrichstraße 22
6520 WORMS
Tel. 06241/6757-6758

6700 Ludwigshafen

MKV Computermarkt
Bismarck-Zentrum
6700 Ludwigshafen
Telefon 0621-525596

6720 Speyer

MKV Computermarkt
Gilgenstraße 4
6720 Speyer
Telefon 06232-77216

6750 Kaiserslautern

C.O.S
Computer Organisation GmbH
6750 Kaiserslautern
Telefon (0631) 92051

Gotthold Bürocenter - Ihr Partner für's Büro
Jac. Gotthold Nacht KG · Kaiserslautern · Im Altstadtparkhaus · Tel. 63071

6800 Mannheim

GAUCH+STURM
Computersysteme + Textsysteme
6800 Mannheim 24
Casterfeldstraße 74-76
☎ (0621) 850040 · Teletex 6211912

Computer-Center
am Hauptbahnhof GmbH
L 14, 16-17
6800 Mannheim 1
Tel. (0621) 20983/84

6900 Heidelberg

JACOM COMPUTERWELT
Hardware · Software
Schulung · Service
Mönchhofstraße 3 · 6900 Heidelberg
Telefon 06221/410514-550

HEWLETT PACKARD
IBM
Schneider
ATARI
HARDWARE
SOFTWARE
SCHULUNG
SERVICE
HEIDELBERGER
COMPUTER CENTER
Bahnhofstr. 1 / Tel. 06221 / 27132

7000 Stuttgart

SCHMITT COMPUTERSYSTEME
Tübingerstr. 18
7000 Stuttgart

7000 Stuttgart

BNT
COMPUTERFACHHANDEL
Ihr starker Partner in Stuttgart
BNT Computerfachhandel GmbH Marktstr. 48 1. Stock
7000 Stuttgart-Bad Cannstatt in der Fussgängerzone
direkt beim Rathaus Telefon (0711) 55 83 83

7022 L-Echterdingen

Autorisierter ATARI-System-Fachhändler
ATARI ST
Matrai computer
Matrai Computer GmbH
Bernhauser Str. 8
7022 L.-Echterdingen
☎ (07 11) 79 70 49

7030 Böblingen

Verkauf – Service – Software
mca
Norbert Hlawinka
Sindelfinger Allee 1
7030 Böblingen
Tel. 0 70 31 / 22 60 15
COMPUTER CENTER

7100 Heilbronn

Unser Wissen ist Ihr Vorteil
Walliser & Co.
Mönchseestraße 99
7100 Heilbronn
Telefon 07131/60048

Seel's **Computer-Welt**
Am Wollhaus 6
7100 Heilbronn
Tel. 0 71 31 - 6 84 01 - 02

7101 Flein

der COMPUTERLADEN von
vortex
in der Falterstraße
7101 Flein
Beratung, Service, Verkauf,
Software-Entwicklung
direkt beim Peripherie-Hersteller

ST-Einkaufsführer
aktuell
preiswert
werbewirksam

7150 Backnang

Computer-Fans finden bei uns alles von:
Servicestation
Vertragshändler
Computer-Systeme
Software-Hardware
commodore
Schneider
ATARI
weeske
Das Elektrohaus am Nordring
Potsdamer Ring 10
7150 Backnang
Tel. 0 71 91 15 28

7410 Reutlingen

Computer-Shop
Werner Brock
autorisierter Fachhändler f. ATARI, Schneider, Oki
Federnseestr. 17 · 7410 Reutlingen
Telefon: 0 71 21 / 3 42 87

7450 Hechingen

SRE
Gesellschaft für Datenverarbeitung mbh
Computer · Drucker
Zubehör · Fachliteratur
Schloßplatz 3 · 7450 Hechingen
Telefon 0 74 71 / 1 45 07

7475 Meßstetten

Ihr ATARI-Systemhändler im Zollern-Alb-Kreis
HEIM + PC-COMPUTERMARKT
HARDWARE · SOFTWARE · LITERATUR
SCHEURER
ATARI COMMODORE CUMANA DATA BECKER
MULTITECH RITEMAN SCHNEIDER THOMSON
7475 Meßstetten 1 · Hauptstraße 10 · 0 74 31 / 6 12 80

7480 Sigmaringen

SOFT&EASY
COMPUTER GMBH
Rapp Gassie
7480 Sigmaringen
Tel 0 75 71 / 1 24 83

7500 Karlsruhe

papierhaus erhardt
Am Ludwigsplatz · 7500 Karlsruhe
Tel. 0721-1608-16

MKV GMBH
Rüppurer Str. 2d
7500 Karlsruhe
Telefon 0721-84613

7530 Pforzheim

DM Computer GmbH
Hard- & Software
Durlacherstr. 39
Tel. 0 72 31 - 1 39 39
7530 Pforzheim
Telex 783 248

7600 Offenburg

FRANK LEONHARDT ELECTRONIC
Ihr Fachgeschäft für Microcomputer · Hifi · Funk
In der Jeuch 3
7600 Offenburg
Telefon 07 81 / 5 79 74

7640 Kehl/Rhein

CSM
Computer-Software-Marketing
eigener Service · eigene Software
Badstrasse 12
Tel. 0 78 07 / 8 22
Telex: 752 913
7607 NEURIED 2
Filiale:
Hauptstrasse 44
Tel. 0 78 51 / 18 22
7640 KEHL/RHEIN
ELEKTRO-MÜNTZER GmbH

7700 Singen

U. MEIER
Computersysteme
7700 Singen-Htwl.
Am Posthalterswäldle 8
Telefon 0 77 31 - 4 42 11

7730 VS-Schwenningen

BUS BRAUCH & SAUTER COMPUTER TECHNIK
Villinger Straße 85
7730 VS-Schwenningen
Telefon 0 77 20 / 3 80 71 - 72

7750 Konstanz

ATARI ★ PC's ★ SCHNEIDER
computer - fachgeschäft
rösler
Rheingutstr. 1 · ☎ 0 75 31 - 2 18 32

7800 Freiburg

CDS EDV-Service GmbH
Windausstraße 2
7800 Freiburg
Tel. 07 61 - 8 10 47

8500 Nürnberg

ATARI

Der ATARI - Systemfachhändler in Nürnberg

Softwareberatung und Entwicklung

Computer • rucker • Zubehör • Fachliteratur

hib

HIB Computer GmbH
Äußere Bayreuther Str. 72
8500 Nürnberg
T4el.:0911/515939

8520 Erlangen

Ihr Partner
IN EDV-FRAGEN

Wir führen prof.
HARD-& SOFTWARE
für
IBM PC/XT/AT
APPLE II+/e
ATARI 520+1040
AMIGA

☎ 09131/22600

ALPHATRON

COMPUTERSYSTEME & SOFTWARE ENGINEERING
marco hildebrandt
luitpoldstraße 22 8520 erlangen

Computerservice Decker

Meisenweg 29 - 8520 Erlangen
Telefon 09131/42076

HAAS
Büro+Computer

8520 Erlangen
Dresdener Str. 5
Friedrichstr. 9
Tel.: 09131/12010

Zimmermann elektroland

8520 Erlangen
Nürnberger Straße 88
Tel. (09131) 34568

8500 Nürnberg
Hauptmarkt 17
Tel. (0911) 20798

8600 Bamberg

BÜRO-ZENTRUM
A+R KUTZ

Bamberg · Tel. 0951/27808-09

8700 Würzburg

HALLER GMBH
Fachgeschäft für
Mikrocomputer
Büttnerstraße 29
8700 Würzburg
Tel. 0931/16705

8700 Würzburg

SCHOLL
BÜROTEAM

Hardware · Software
Service · Schulung

computer center
am Dominikanerplatz
Ruf (0931) 50488

Müller-Anwendersoftware
Versandhandel für PC-Software
Textverarb., Graphik, Bucht.,
Spiele PD-Software erklärt u. a.
Matterstockstr. 23a · 8700 Würzburg
Tel. 0931/282326

8720 Schweinfurt

Uhlenhuth GmbH

Computer + Unterhaltungselektronik

Albrecht-Dürer-Platz 2
8720 Schweinfurt
Telefon 09721/65 2154

8900 Augsburg

Adolf & Schmoll Computer

Schwalbenstr. 1 · 8900 Augsburg
Tel. (0821) 528533 oder 528087

Wir sind außerdem autorisierte Service-Fachwerkstatt für:

Schneider
commodore
ATARI

ATARI Systemfachhändler

PROFI COMPUTER STUDIO

KARSTADT Aktiengesellschaft
Bürgermeister-Fischer-Str. 6-10
8900 Augsburg · Tel. (0821) 3153-416

SCHMITT
COMPUTERSYSTEME

Frauentorstr. 22
8900 Augsburg
✆ 0821-154268

8940 Memmingen

EDV-Organisation
Hard- + Software
Manfred Schweizer KG
Ulmer Str. 2, Tel. 08331/12220
8940 Memmingen

Österreich

A-1020 Wien

W + H.
Computerhandel Ges m.b.H.

Förstergasse 6/3/2 · 1020 Wien
Tel. 0222-350968

A-1040 Wien

Ihr ST-Fachhändler in Wien

Computer-Studio
Wehsner Gesellschaft m.b.H.

1040 Wien - Paniglgasse 18-20
Telefon 0222-657808, 658893

A-8010 Graz

zupan

EINSTEIGEN
DABEISEIN
VORANKOMMEN

A-8010 Graz, Mandellstraße 23, Tel. (0316) 70 28 40, 70 30 23
Tlx. 03 2534 zupan a

Schweiz

CH-2503 Biel

UE
URWA Electronic

Ihr ATARI ST Spezialist in der Schweiz.
032/254553

Lindenweg 24, 2503 Biel

CH-3084 Bern

Computer Corner

Ihr ATARI ST-Partner in Bern

Hard-Software
Midi-Schulung

Beratung
Service

Seftigenstr. 240
Zentrum Wabern

3084 Bern-Wabern
Tel. 031/54 51 00

EUMEL

– Ein neues Betriebssystem für den Atari ST

TEIL 3

```
..................... selforganization ..................... Zeile 8
#type ("prop")##limit (14.0)#
#linefeed (2.0)#
#pagenr("%",1)#
#head#
#center#- % -

#end#
.............................. 1. Einleitung .................. Zeile 73
#center##on("b")##on("u")#1. Erkundigung (1914-1920)#off("b")##off("u")#

Die Schriften Thomas Manns dieser Zeit sind geprägt von Konservat-
ismus und Patriotismus auf der Basis einer humanistischen Weltan-
schauung. Geistig beeinflußt von Nietzsche, Hegel und Schoppenhau-
er, später von Ortega y Gasset's "Der Aufstand der Massen" - in
Deutsch erschienen 1931 - , von Ernst Troeltsch' "Naturrecht und
Humanität in der Weltpolitik" (7) und Zeitgenossen wie Paul Amann
und Ernst Bertram, kann man gerade an d ......................... Zeile 1
beiden Personen den geistig-politischen
vollziehen. Der Briefkontakt mit Ernst   25.04.87        "testtext"
die frühen politischen Ansichten Manns   25.04.87        "testtext.p"
Wandlung - neben seinen brieflichen Äuß  25.04.87        "schöntext"
sache, daß er in den Jahren ab 1914 sic  25.04.87        "Inhaltsverzeichnisse
```

Bild 1: Drei Texte parallel bearbeitet – kein Problem mit EUMEL.

EUMEL für Programmierer

Im letzten Teil unserer Einführung in das neue Multitasking Betriebssystem EUMEL soll es zunächst um die Programmieroberfläche gehen. Bereits im vorherigen Heft wurde der Editor vorgestellt, der jedoch über einige – bisher noch unerwähnte – Funktionen verfügt, die für den Programmierer interressant sind. Da ist zunächst die Möglichkeit, mit nahezu unendlich vielen Fenstern parallel zu arbeiten (Bild 1). Jedes neue Fenster kann an einer beliebigen Stelle auf dem Bildschirm geöffnet werden: Nach der doppelten Betätigung der Escape Taste (ESC) hat man die Monitorebene vor sich und gibt »edit („dateiname“)« an; dann erscheint an der aktuellen Cursorposition ein zusätzliches Textfenster. Ein Wechseln zwischen den Fenstern ist mit ESC-w jederzeit möglich. Leider fehlt die Möglichkeit, ein Fenster nachträglich in seiner Position zu verändern.

Der Programmierer kann mit der Fenstertechnik in einem Bildschirmausschnitt seinen Sourcecode in ELAN erstellen, in einem zweiten Fenster die Auswirkungen des gestarteten Programms verfolgen und in dem Notizbuch eventuelle Fehlermeldungen ablesen. Das Notizbuch ist nichts anderes als eine ständig präsente Datei, die mit ESC-n aufgerufen wird und in der alle Anmerkungen des ELAN Compilers vorliegen. Wer die schnell vorbeihuschenden Fehlermeldungen anderer Compiler kennt, wird dieses Detail zu schätzen wissen.

Die Programmentwicklung unter EUMEL verläuft in der Regel wie folgt: zunächst wird ein ELAN Programm mit dem Editor erstellt, wobei natürlich alle Möglichkeiten der Textverarbeitung (Blöcke verschieben, Suchoperationen etc.) zur Verfügung stehen. Im nächsten Schritt wird das erstellte Programm mit dem Befehl »run („dateiname“)« dem ELAN Compiler übergeben. Dieser nimmt eine Überprüfung in zwei Durchgängen vor, schreibt eventuelle Fehler in das Notizbuch und startet das Programm, – vorausgesetzt es ist fehlerfrei (Bild 2).

Sofern ein einmal korrekt ablaufendes Programm häufiger verwendet werden soll, gibt es die Möglichkeit, das Programm als Paket zu insertieren, d. h. in der Benutzertask dauerhaft zur Verfügung zu stellen (ein „Paket“ als Sammlung von Prozeduren ist für ELAN ungefähr das, was bei MODULA 2 das „module“ darstellt). Der Befehl »insert („dateiname“)« erledigt dies völlig automatisch. Auch hier wird allerdings zunächst der ELAN Compiler vorgeschaltet und das Programm überprüft. Nun steht das einmal insertierte Programm als Monitorbefehl jederzeit (ohne Nachladen von Diskette o. ä.) zum Aufruf bereit.

Dieses Prinzip wird auch bei kommerziellen Anwenderprogrammen verwendet. Die Dateiverwaltung EUDAS von Thomas Berlage wird z.B. als Sourcefile geliefert und mit dem hier beschriebenen Verfahren insertiert. Der Start von EUDAS verläuft dann genauso wie der des Editors: »eudas („dateiname“)«.

Um die redundante Speicherung von Paketen zu vermeiden, existiert die äußerst einfallsreich konzipierte Option, die in Vatertasks insertierten Programme auch allen Sohntasks bereitzustellen. Sollen mehrere Benutzer auf die Dateiverwaltung zurückgreifen, so reicht es aus, eine Vatertask mit EUDAS einzurichten und somit steht das gesamte Dateiprogramm für alle Söhne bereit. Desweiteren erlaubt es das Paketprinzip von ELAN, modular zu programmieren. Große Programme können in kleine „Häppchen“ aufgeteilt werden, die jeweils einzeln bearbeitet und überprüft werden (Bild 3).

Und noch mehr: Die vom Betriebssystem bereitgehaltenen Standardpakete dürfen ebenfalls unter ELAN in einem Anwenderprogramm verwendet werden (Bild 4). So ist es nicht nur möglich, Monitorbefehle vom Editor aus einzugeben, vielmehr kann auch der Editor aus einem Benutzerprogramm heraus aufgerufen werden. Allgemein ausgedrückt heißt dies, daß alle Monitorbefehle nichts anderes als insertierte Pakete darstellen, die in ELAN Programme einbindbar sind.

```
      eof type    =  7;

    TEXT VAR zeile, taste, taste 2, s1, s2;
    INT VAR p, i, t1, t2;


    PROC bestimme taste :

      p := pos (zeile, " ", 3);
      scan ( subtext (zeile, 1, p) );
      next symbol (s1, t1);
      IF t1 = text type

          gib kommando :
          insert ("tastenbelegung")
      281

    Keine Fehler gefunden,  1346 B Code, 258 B Paketdaten generiert


           ******  ENDE DER UEBERSETZUNG  ******
```

Bild 2: Hier wurde ein Programm insertiert. Der EUMEL gibt an, wieviele Paketdaten er generiert hat.

```
............................... list ....................... Zeile 1
........................ eumel to xywrite .................. Zeile 1
PACKET eumelanpassung DEFINES eumel to xywrite:

PROC eumel to xywrite:
     eumel to xywrite(last param);
END PROC eumel to xywrite;

PROC eumel to xywrite (TEXT CONST name):
TEXT VAR a := "n";
IF   exists (name)
THEN FILE VAR f :: sequential file (modify, name);
     sonderzeichen umcodieren;
     putline("Die Datei wurde umcodiert.");
     put("Soll sie auch umgebrochen werden?  (j/n)"); inchar(a);
     IF   a = "j"
     THEN umbrechen FI
ELSE errorstop ("Diese Datei gibt's nicht!")
FI;
toline(f,1);
last param(name);
line.
```

Bild 3: Ausschnitt aus einem Filterprogramm. Definiert wird das Paket „eumel to xywrite".

```
................................................................ Zeile 1
      a..............
      bits...........
      text...........
      pcbandinitcontrol
      dataspace......
      basictransput..
      bool...........
      integer........
      errorhandling..
      real...........
      datehandling...
      commanddialogue
      thesaurushandling
      localmanager...
      patternmatch...
      filehandling...
      elandointerface
      scanner........
      screendescription
      tastenverwaltung
      editorpaket....
      editorfunctions
```

Bild 4: Eine Liste der vorhandenen Pakete mit dem Befehl „packets".

ELAN

Die Programmiersprache ELAN (ELementary LANguage) ist ein Abkömmling der ALGOL68 Familie und erinnert in seinen Strukturen stark an PASCAL oder MODULA 2. Als elementare Datentypen stehen Integer- (INT) und Realzahlen (REAL), Zeichenketten (TEXT) und Wahrheitswerte (BOOL) bereit. Datenobjekte müssen, wie in fast jeder Hochsprache, deklariert werden. ELAN kennt natürlich keine GOTO-Befehle (wozu auch?) sondern die oben erwähnten Pakete und Refinements. Letztere sind quasi Unterprogramme mit eigenem Namen, die die „top down" Programmierung ermöglichen. Ein kleines Programm könnte etwa wie folgt aussehen:

Tabelle 1 – Ein Beispielprogramm unter ELAN

INT VAR a,b,x
Einlesen von a und b;
Vertauschen von a und b;
Neue Werte ausgeben.

Einlesen von a und b:
  get (a);
  get (b).

Vertauschen von a und b:
  x := a;
  a := b;
  b := x.

Vertauschte Werte ausgeben:
  put (a);
  put (b).

Tabellenende

Refinements können verschachtelt sein, – damit reicht es aus, nur Teile einer umfassenden Problemlösung zu bearbeiten. Daß die Programmstruktur mit diesem Prinzip äußerst übersichtlich und dokumentationsfreundlich wird, ergibt sich von selbst. Schleifen und Abfragen können leicht konstruiert werden. Eine kleine Auswahl zeigt Tabelle 2.

Tabelle 2 – Schleifen und Abfragen unter ELAN

WHILE ... REPEAT
REPEAT ... UNTIL
FOR ... FROM ... UPTO/DOWNTO ... REPEAT
IF ... THEN ... ENDIF
IF ... THEN ... ELSE ... ENDIF

Tabellenende

```
...................................f..................................... Zeile 12
 ! " # $ % & ' ( ) * + , - . / 0 1 2 3 4 5 6 7 8 9 : ; < = > ? @ A B C D E
 F G H I J K L M N O P Q R S T U V W X Y Z [ \ ] ^ _ ` a b c d e f g h i j
 k l m n o p q r s t u v w x y z { | } ~ ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ?
 ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ?
 ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? Ä Ö Ü ä
 ö ü k - #   ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ß # ? ?
 ?

...................................t..................................... Zeile 4
INT VAR i;
FOR i FROM 32 UPTO 255 REP
type (code (i) + " ")
END REP
```

Bild 5: Vier Zeilen Programmcode sind erforderlich, um den Zeichensatz in ein Textfenster auszugeben. Mehr nicht!

Es würde zu weit führen, alle Sprachelemente aufzuzählen, – für weitere Informationen ist eine Literaturliste am Ende des Artikels abgedruckt. Insgesamt läßt sich jedoch sagen, daß ELAN eine auch für Programmierlaien leicht zu erlernende Sprache ist, die das strukturierte Abarbeiten von Problemen gestattet und die in den Betriebssystemkern eingebunden werden kann (Bild 5). Dabei kommt die Professionalität in keinem Punkte zu kurz: Auch kommerziell vertriebene Anwenderprogramme sind in der Regel in ELAN abgefaßt.

Kritikpunkte sind dennoch aufzuführen. Real-Time Anwendungen sind derzeit nicht machbar. Dies liegt u. a. daran, daß – zweitens – die EUMEL Übertragung auf den Atari ST nicht im maschinennahen „Native-Code" vorgenommen wurde. In puncto Geschwindigkeit ist deshalb der EUMEL auf dem Atari ST nicht der schnellste. Als Vergleichsrechner stand uns der BICOS 286 zur Verfügung, der mit dem Intel-Prozessor 80286 und einer Taktfrequenz von immerhin 16 MHz arbeitet. Wir mußten feststellen, daß bei verschiedenen Benchmarks der Atari ST höchstens halb so schnell lief wie der BICOS 286. Wer als Einzelbenutzer mit dem Atari arbeitet, wird davon nur wenig merken, – Schwierigkeiten gibt es erst bei parallel laufenden Aufgaben.

Der Hamster in der Schule

Zu den mitgelieferten Programmpaketen unseres Testexemplars gehört u. a. der Hamster. Das ist eine kleine Kunstwelt, die für die Programmierausbildung geschaffen wurde und die sich vorzüglich zum Schuleinsatz eignet. Herbert, der Hamster, bewegt sich auf einer gekachelten Landschaft, die einige Körner aber auch Hindernisse enthält (Bild 6). Herbert kann sich in seiner Umwelt frei bewegen und Körner in seiner Backentasche ablegen. Um die Landschaft zu erfassen, kann Herbert verschiedene Tests vornehmen: er fragt, ob die vor ihm liegende Kachel frei ist, ob ein Korn in der Umgebung liegt und wieviele Körner sich bereits in seinen Backentaschen befinden. Komplexere Suchstrategien lassen sich in einem Hamsterprogramm formulieren und erarbeiten.

Auch wenn diese Mischung aus Logo und Robotersimulation sicherlich nicht jedermanns Sache ist: zwei Dinge sind dennoch bemerkenswert. Zum einen zeigt der Hamster wie auch mit ELAN eine benutzerfreundliche Menüsteuerung implementiert werden kann. Auf Bild 7 sehen Sie, daß alle Befehle des Hauptmenüs mit den Cursortasten selektierbar sind, – man kann also mit ELAN eine völlig neue Benutzerumgebung schaffen, ohne daß dies zu Lasten der Leistungsfähigkeit geht. Zum anderen soll das Beispiel des Hamsters zeigen, daß EUMEL insbesondere für den Schuleinsatz einige herausragende Einsatzmöglichkeiten bietet. Am Hamster kann der Algorithmusbegriff für das strukturierte Lösen von Problemen eingeführt werden; Hemmschwellen gegenüber dem Computer verschwinden durch den spielerischen Charakter des Programms. Darüberhinaus ist das Programmpaket SCHULIS speziell für den Unterricht vorgesehen. Mit dem Modellcomputer MOCO können grundlegende Eigenschaften von Computern mit minimalen Grundkenntnissen vermittelt werden. Für den naturwissenschaftlichen Unterricht in der Oberstufe exisitert SIMSEL, ein Werkzeug, mit dem sich gekoppelte Differentialgleichungen behandeln lassen. Und schließlich gehört zum SCHULIS Lieferumfang ein komplettes Verwaltungssystem für Sekretärinnen, Lehrer und Schulleitung. Mit ihm sind organisatorische Routineaufgaben auch ohne Informatikvorkenntnisse schnell in den Griff zu bekommen.

Simulationen mit DYNAMO

Ein anderes Paket, das uns für EUMEL geeignet erschien, und in der Forschung Anwendung findet Es ist die Simulationssprache DYNAMO, die von Jay Forrester in den sechziger

```
.......................... xywrite to eumel ............................ Zeile 25

PROC xywrite to eumel:
     xywrite to eumel(last param);
END PROC xywrite to eumel;

PROC xywrite to eumel (TEXT CONST name):
TEXT VAR a := "n";
IF   exists (name)
THEN FILE VAR f :: sequential file (modify, name);
     sonderzeichen umcodieren;
     putline("Die Datei wurde umcodiert.");
     put("Soll sie auch umgebrochen werden?  (j/n)"); inchar(a);
     IF   a = "j"
     THEN umbrechen FI
ELSE errorstop ("Diese Datei gibt's nicht!")
FI;
toline(f,1);
last param(name);
line.

sonderzeichen umcodieren:
putline("Datei wird von Compaq- auf EUMEL-Zeichensatz umcodiert");
```

Bild 6: Herbert der Hamster auf Körnersuche.

```
Herbert der Hamster

1 ..... Informationen ueber den Hamster

2 ..... Hamster von Hand in einer Landschaft steuern

==> 3 ..... neue        Landschaft    erstellen
    4 ..... vorhandene  Landschaft    ansehen oder veraendern
    5 ..... vorhandene  Landschaften  drucken

    6 ..... neues       Programm      erstellen
    7 ..... vorhandenes Programm      ansehen oder veraendern
    8 ..... vorhandene  Programme     drucken

    9 ..... Hamster nach Programm in einer Landschaft laufen lassen

    w ..... Weitere Funktionen
    z ..... Zurueck in die 'gib kommando' - Ebene
```

Bild 7: Menüsteuerung mit Cursortasten: auch unter EUMEL keine Schwierigkeit.

Jahren am Massachusetts Institute of Technology entwickelt wurde. DYNAMO erlaubt als nicht-prozedurale Sprache die Simulation von Systemen, in denen Veränderungen kontinuierlich entstehen. Dabei kann man vorhandene mathematische Gleichungen fast ohne Modifikationen dem Compiler vorsetzen und erhält die in Bild 8 dargestellten Diagramme. DYNAMO wird vor allem bei ökologischen, gesellschaftlichen, wirtschaftlichen und technischen Simulationen eingesetzt und zeigt erneut, wie einfach mit ELAN auch eine Kunstsprache geschaffen werden kann. Dies gilt auch für LISP, die Sprache der künstlichen Intelligenz, die John McCarthy mit ELAN realisiert hat und die, wie das Hamsterprogramm kostenlos ist.

Statistik mit DASYS

Schließlich haben wir das Datenauswertungssystem DASYS untersucht Das Programm ist keine Tabellenkalkulation im herkömmlichen Sinne, sondern ein Zusatzpaket, das auf Tabellen zurückgreift, die mit dem EUMEL Editor erstellt wurden. DASYS beherrscht die wichtigsten Grundfunktionen der Statistik, dazu zählen zunächst einfache Kenngrößen wie die Berechnung von Minimum und Maximum, Spannweite, Median, Summe, arithmetisches Mittel, Standardabweichung und Varianz. Häufigkeitsauszählungen bei stetigem und diskretem Merkmal gehören schon zu den komplexeren Funktionen. Dies gilt auch für die Erstellung einer Korrelationsmatrix zur Berechnung des Pearsonschen Stichproben-Korrelationskoeffizienten. Auch für die Lineare Regression können Taschenrechner und Bleistift beiseite gelegt werden, – DASYS erstellt automatisch eine vollständige Regressionsgleichung aus vorhandenem Datenmaterial. Sicherlich kann ein Statistiksystem wie SPSS von DASYS bei hochkomplexen Anforderungen nicht ersetzt werden, aber für den kleinen Bedarf stellt das Paket für 200 DM durchaus eine Alternative dar.

Preise

Damit sind wir bei den Preisen angelangt. Das Betriebssystem EUMEL kostet in der Multiuser-Version 750 DM (alle Angaben ohne Mwst.), für Schulen und Hochschulen die Hälfte. Dazu kommen die erforderlichen Programmpakete. Die Textverarbeitung ist für 850 DM zu haben; wer alte Dateien vom Atari ST oder IBM Clone bearbeiten möchte, muß 200 DM für das Transferpaket von Frank Klapper bezahlen. Auch für die Vernetzung mehrerer EUMEL Stationen untereinander wird ein Aufpreis erhoben: 200 DM kostet die notwendige Software. Auf den ersten Blick mag dies für den privaten Anwender viel Geld sein, – wenn man aber die Leistungsfähigkeit des EUMELs berücksichtigt, ergibt sich für die Minimalkonfiguration eine Software-Investition von 1.600 DM. Soviel kostet ein sehr gutes Textprogramm für die MS-DOS Welt, zumeist ohne Netzwerkfähigkeit und die hier vorgestellten Optionen. Wenn wir EUMEL mit vergleichbaren Systemen konfrontieren, zeigt sich schnell, daß dieses Betriebssystem relativ günstig zu haben ist.

Für wen ist EUMEL empfehlenswert?

Zunächst sei an den Anwender gedacht, der selbst programmieren will oder muß, – vielleicht schon deshalb, weil keine Standardprogramme erhältlich oder Spezialaufgaben zu erledigen sind. Textverarbeitung, Dateiverwaltung und Tabellenkalkulation sollen integriert sein und die Programmiersprache soll hochkomplexe Anforderungen mit möglichst geringem Aufwand erledigen können. In diesem Fall ist EUMEL ohne Zweifel die ideale Softwarelösung. Mit einem Kostenaufwand von rd. 6.000 DM (Atari 1040 ST, Festplatte, Drucker und EUMEL System mit Textverarbeitung) zuzüglich der Kosten für Dateiverwaltung und Tabellenkalkulation liegt man ungefähr bei dem, was für einen IBM Clone und ein integriertes Softwarepaket zu zahlen sind, die Regel nicht annähernd die Flexibilität und Leistungsfähigkeit, geschweige denn den Multitasking und Multiuserfähigkeiten des EUMELs erreichen.

Wenn auch ein „Freak" vielleicht nur am Rande von den Multitaskingeigenschaften des EUMELs Gebrauch macht, – so wird z. B. Schreibbüro auf diese nicht verzichten wollen. Als zusätzliche Investitionen sind hier Terminals (rd. 1.000 DM) und Laserdrucker (ab 6.000 DM) aufzuführen. Zum Preis eines Kleinwagens erhält man eine EUMEL Station mit mehreren Arbeitsplätzen, die mit Ausnahme der Graphikintegration alles bietet, was zum Buchdruck erforderlich ist. Eine vergleichbare Konstellation wäre etwa ein IBM-AT System mit Textprogramm und Laserdrucker, bei dem man allerdings auf mehrere Arbeitsplätze und das Multitasking verzichten müßte.

Auch Großfirmen sind mit EUMEL sicherlich gut bedient. In diesem Fall wird man allerdings dem BICOS 386 (mit Intels 80386 Prozessor) gegenüber dem Atari ST den Vorzug geben. 10 Terminals werden an eine Station mit Zentralrechner angeschlossen und beliebig viele EUMEL-Knotenpunkte können mit der Netzsoftware verbunden werden. Über das Netz verschickt man alle notwendigen Daten mit nur

```
      run

X=X,Y=Y,Z=Z(-10,10)
      -10.0      -5.0      0.0       5.0       10.0xyz
 0.0  .__________.__________x__________.__________.yz
 2.0  .          .          y x        .          .z
 4.0  .          .          y   x      .          .z
 6.0  .          .          y     x    .          .z
 8.0  .          .          y       x  .          .z
10.0  .__________.__________y_________x.__________.z
12.0  .          .          y          x          .z
14.0  .          .          y          . x        .z
16.0  .          .          y          .   x      .z
18.0  .          .          y          .     x    .z
20.0  .__________.__________zy_________._______x__.
22.0  .          .          zy         .         x.
24.0  .          .          zy         .          x.
26.0  .          .          z y        .          x.
28.0  .          .          z y        .         x.
30.0  .__________.__________z__y_______._______x__.
32.0  .          .          z  y       .     x    .
```

Bild 8: Eine graphische Ausgabe unter DYNAMO.

einem Befehl, der Empfänger holt sich mit „get my mail" die Eingangspost jederzeit ab. Auch der Hardwareeinsatz ist dann effektiver: Den Drucker am Knotenpunkt ergänzt ein zentral aufgestellter Laserdrucker, der über das Hausnetz alle Texte mit hohen Qualitätsansprüchen empfängt. Ein Beispiel für den Großeinsatz ist die Universität Bielefeld, an der der EUMEL u.a. entwickelt wurde. Dort sind an Instituten, Lehrstühlen und auch im öffentlichen Zugang rund 400 Terminals mit fast 50 Knotenpunkten installiert. Jede EUMEL Station verfügt über einen Standarddrucker und im Hochulrechenzentrum übernimmt der Laserdrucker die Ausgabe für alle Benutzer, die sich durch eine Zugangsberechtigung ausweisen können.

Ein weiteres Beispiel für den sinnvollen Einsatz des Betriebssystems EUMEL ist der Schulbereich. Nicht nur die Schulsoftware SCHULIS, sondern auch die strukturierte Programmiersprache ELAN und der Multiuserbetrieb sind Vorzüge, die in dieser Preisklasse kein anderes System zu bieten hat. Daß der EUMEL Einsatz nicht nur auf den Informatikunterricht beschränkt ist, haben wir bereits gesehen. Auch der naturwissenschaftliche Unterricht und die Schulverwaltung kommt nicht zu kurz. Das haben inzwischen rund 800 Schulen erkannt, die bereits erfolgreich EUMEL in Ausbildung und Lehre einsetzen.

Unsere Serie konnte selbstverständlich nur einige, wenige Ausschnitte des neuen Betriebssystems für den Atari ST vorführen. EUMEL hat bis jetzt in der Computerwelt noch nicht die Bedeutung erlangt, die ihm von der Leistungsfähigkeit her zustände. Das mag daran liegen, daß das Betriebssystem nicht in dem Maße kommerziell vermarktet wird, wie dies ein großes Softwareunternehmen, aber nicht die öffentlich-rechtliche „Gesellschaft für Mathematik und Datenverarbeitung" leisten könnte. Es ist bekannt, daß auch mangelhafte Betriebssysteme etablierter Firmen durch viel Public Relations und Sponsortätigkeit durchsetzbar sind. EUMEL glänzt hingegen nur durch seine inneren Qualitäten – und zwar ganz kräftig

Michael Spehr

Literatur zum Weiterlesen:

[1] **Hommel/Jäckel/Jähnichen u. a.**, ELAN-Sprachbeschreibung, Akademische Verlagsgesellschaft Wiesbaden

[2] **Hahn/Stock**, ELAN-Handbuch, Akademische Verlagsgesellschaft Wiesbaden

[3] **Klingen/Liedtke**, Programmieren in ELAN, Teubner-Verlag Stuttgart

[4] **Klingen/Liedtke**, ELAN in 100 Beispielen, Teubner-Verlag Stuttgart

[5] **Craemer**, Das Buch zum DYNAMO Compiler auf dem EUMEL System, Teubner-Verlag Stuttgart

[6] **Danckwerts/Vogel/Bovermann**, Elementare Methoden der Kombinatorik, Teubner-Verlag Stuttgart

[7] **Blau**, Sprachenband ELAN, Metzlersche Verlagsbuchhandlung Stuttgart

[8] **Ambros**, Der Hamster. Programmieren lernen in einer Modellwelt, Metzlersche Verlagsbuchhandlung Stuttgart

True BASIC™

TRUE BASIC ist ein schneller Basic-Interpreter/Compiler mit umfangreichem Wortschatz und komfortablem Editor. Entwickelt wurde es von John G. Kemeny und Thomas E. Kurtz, die schon vor etwa 20 Jahren die Urversion von BASIC erstellt haben.

TRUE BASIC ist ein Basic-Dialekt, der auf verschiedenen Betriebssystemen und Computern lauffähig ist, z. B. auf IBM-Computern und Kompatiblen, Apple Macintosh, AMIGA und auch auf dem ATARI ST. Dadurch können Programme, die auf verschiedenen Rechnern mit TRUE BASIC erstellt wurden, direkt auf einen anderen übertragen werden. Die Anpassung bezieht sich dabei , sofern der jeweilige Computer grafikfähig ist, auf die Grafik und die Fenstertechnik.

Nicht übertragbar sind dagegen die speziellen Systembefehle der einzelnen Rechner, auf die mit speziellen Libraries zugegriffen werden kann.

Der Editor

Eines der wichigsten Bestandteile des Basic-Systems ist der Editor, denn mit diesem Teil verbringt man die meiste Zeit bei der Programmentwicklung. Er besteht aus drei Fenstern (Edit, Command, Output) und einer Menüzeile. Zusätzlich lassen sich die wichtigsten Editorbefehle über Tastencodes aufrufen, so daß man die Maus unbenutzt lassen kann. Sehr wichtig sind die Befehle zum Suchen und Ersetzen von Textpassagen, oder zum Anspringen eines bestimmten Unterprogramms. Außerdem gibt es Möglichkeiten, schnell ans Ende oder den Anfang des Listings zu gelangen. Da TRUE BASIC sowohl mit, als auch ohne Zeilennummern arbeitet, gibt es eine Funktion, die Zeilennummern löscht bzw. einfügt. Nun gibt es Programmierer, die weniger gern ihre Programme so strukturieren, daß Schleifen und Abfragen eingerückt sind. Ihnen wird mit dem Befehl DO FORMAT geholfen, denn dieser nimmt das Einrücken der betreffenden Zeilen nachträglich vor, damit das Listing übersichtlicher ist.

Sehr hilfreich sind auch die Befehle ECHO, DO TRACE und DO XREF. Nach Eingabe von ECHO werden alle Bildschirmausgaben auf den Drucker oder ein File umgelenkt. DO TRACE startet die Fehlerbehandlungsroutinen. Dabei werden Optionen wie STEP, SLOW, FAST, BREAK und unter Umständen noch Bedienungen und Variabeln überprüft. Damit können aus dem Gesamtprogramm leicht bestimmte Bereiche herausgelöst werden, so daß die Fehlersuche einfach zu optimieren ist. Der letzte, aber nicht unwichtigste Befehl ist DO XREF. Er gibt ein mit Zeilennummern versehenes Listing aus und danach eine Crossreferenz-Liste der verwendeten Variablen und Befehle. Dies bedeutet, daß hinter jeder Variablen alle Zeilennummern angegeben werden, in denen sie vorkommt. Dies ist eine große Hilfe bei der Fehlersuche und auch schon bei der Programmentwicklung, weil man alle verwendeten Variablen geordnet vor sich hat.

```
File  Edit  Alter  Run
HANOI.TRU                                  Output
! Towers of Hanoi                          The Towers of Hanoi
!
CLEAR
LET maxcolor = 4

DIM count(0 to 6,3)
OPEN #3: screen .1875,.8125,.2,.8
SET WINDOW 0,4,0,8
PLOT TEXT, AT .5,9: "The Towers of Hano
WINDOW #3

LET disks = 6
LET count(0,1) = disks                  ! All on disk 1
FOR i = 1 to disks
    LET count(i,1) = disks+1-i
NEXT i

SET COLOR mod(0,maxcolor)+1
BOX AREA 0,4,0,1
SET COLOR mod(2,maxcolor)+1
```

1. Bildschirm von True Basic

Strukturbefehle

TRUE BASIC bietet viele Befehle an, die es ermöglichen, klar gegliederte und übersichtliche Programme zu schreiben, wie dies auch in Sprachen wie PASCAL, C und MODULA vorgesehen ist. Daher stehen neben der gewöhnlichen FOR..NEXT-Schleife noch Strukturen wie z.B.: DO WHILE/LOOP oder DO UNTIL/LOOP und verschiedene Variationen zur Auswahl. Damit sind sowohl abweisende als auch nichtabweisende Schleifen programmierbar. Alle Schleifen können mit einem EXIT-Befehl verlassen werden, was jedoch nicht immer der beste

Weg ist. Die IF/THEN-Struktur wurde um ELSE und weitere ELSEIF-Zweige erweitert und ist nicht auf eine Zeile beschränkt. Somit kann bei jeder Bedingung eine beliebige Anzahl von Befehlen ausgeführt werden. In Anlehnung an Sprachen wie PASCAL u. a. wurde auch der Entscheidungsbefehl CASE aufgenommen:

```
SELECT CASE Ausdruck
  CASE Konstante1 Anweisung(en)
  CASE Konstante2 Anweisung(en)
  CASE ELSE Anweisung(en)
END SELECT
```

Sehr gut gelungen sind die Funktions- und Prozeduraufrufe, die auch Rekursionen zulassen und an PASCAL angelehnt sind. Damit können Programme sehr einfach modernisiert werden, wobei dann im Hauptprogramm fast nur noch die Prozeduraufrufe stehen.

1) DEF Func(x,y,...) = Ausdruck
   Func(x,y,...)
2) SUB Sub(x,y,...)
   CALL Sub(x,y,...)

DEF FUNC und SUB sind Funktions- bzw. Prozeduraufrufe mit Parameterübergabe. Das heißt, daß der aufrufende Befehl (z.B. CALL Sub(5,3)) Werte (hier: 5,3) an das Unterprogramm übergibt, die dann von diesem weiterverarbeitet werden, ohne daß die Variablen des aufrufenden Programmes verändert werden. Es ist auch möglich, diese Aufrufe mit Rückgabeparametern zu versehen.

Funktionen und Prozeduren können auch extern definiert werden. Dies bedeutet, daß man oft benutzte Prozeduren in ein separates File schreibt und sie dann vom Hauptprogramm aus mit dem Befehl LIBRARY einbindet. Diese Prozedurfiles können auch von mehreren Programmen verwendet werden. Auf die gleiche Weise werden die systemspezifischen Befehle der jeweiligen Rechner und die Erweiterungen nutzbar gemacht. Möglich ist es auch, externe Prozeduren in der Sprache 'C' oder in Assembler zu schreiben und von TRUE BASIC aus aufzurufen.

Mathematische und trigonometrische Funktionen

Die mathematischen Funktionen enthalten neben ABS, EXP, INT, LOG, LOG2, LOG10, MAX, MIN, RND, SGN, SQR noch Befehle zum Runden (ROUND) und Abschneiden (TRUNCATE) von Nachkommastellen. Hinzu kommen noch MODULO, DIVIDE und REMAINDER, die beim Teilen von Zahlen den ganzzahligen Anteil oder/und den Rest angeben. Die trigonometrischen Funktionen bestehen neben SIN, COS, TAN noch aus ATN, PI, DEG, RAD und ANGLE. Weitere Funktionen (z.B. die Areafunktionen usw.) sind in den Libraries definiert, die bei Bedarf eingebunden werden können.

Sehr praktisch ist es, daß Winkelangaben im gebräuchlichen Bogenmaß und auch im Grad eingegeben werden können. Dies erspart bei manchen Anwendungen die Umrechnung.

Sehr leistungsfähig sind die eingebauten Matrix-Operationen. Mit einem einzigen Befehl kann eine ganze, mehrspaltige Matrix eingelesen werden. Matrixaddition, -Subtraktion, -Multiplikation und -Inversion stellen ebenfalls kein Problem mehr dar.

Stringfunktionen

Einige spezielle Stringoperationen erweitern das 'Standard-BASIC': LCASE und UCASE wandeln Buchstaben in Klein- bzw. Großbuchstaben; TRIM, RTRIM und LTRIM entfernen Leerzeichen, die an verschieden Stellen des Strings stehen; REPEAT kopiert einen String beliebig oft in einen anderen; USING erlaubt die formatierte Ausgabe von Zeichen, Strings und Zahlen; POS gibt den Ort an, an dem ein Teilstring in einem anderen String steht. Auffallend für Basic-Programmierer ist, daß die Befehle LEFT$, RIGHT$ und MID$ fehlen. Bei TRUE BASIC gibt es dafür nur die Möglichkeit, hinter dem Stringnamen den Anfangs- und Endwert anzugeben, um einen Teilstring zu erhalten. Dies erfordert ein gewisses Umdenken beim Schreiben der Programme.

```
library "d:\libs\system.trc"
declare public control, int_in, intin

call pokew(control,72)
call pokew(control+2,6)
call pokew(control+4,1)
call pokew(control+6,0)
call pokew(int_in,40)
call pokew(int_in+2,40)
call pokew(int_in+4,0)
call pokew(int_in+8,150)
call pokew(int_in+10,150)

for i=1 to 10
  call s_aes
next i

call pokew(contrl,15)
call pokew(contrl+2,0)
call pokew(contrl+6,1)
call pokew(intin,3)

call s_vdi
plot 0,0;1,1

end
```

VDI-Aufruf in True Basic

Dateien

Es werden drei Typen von Dateien unterstützt: TEXT, RECORD und BYTE. TEXT-Dateien sind die 'normalen' Dateien, die mit INPUT und PRINT beschrieben und gelesen werden. RECORD und BYTE sind platzsparende Dateien, die mit READ und WRITE behandelt werden. Dateien können mit einer Schleife eingelesen werden, welche die Endbedingung enthält: DO WHILE MORE #1 ... bzw. DO WHILE MORE DATA ... Einige weitere Befehle erleichtern die Behandlung von Dateien. TRUE BASIC kommt somit den Programmierern entgegen, die sich bisher gegen die schlechten Möglichkeiten und die dadurch notwendigen Mühen und Einschränkungen ausgesprochen haben. TRUE BASIC hat auch hier einiges von anderen Sprachen übernommen. Zusätzlich sind noch Befehle dazugenommen worden, damit das Arbeiten möglichst einfach ist. Sehr hilfreich ist dabei der Befehl ASK, der viele Informationen über eine Datei geben kann (Status, Organisation, Größe, Zeigerstand, Recordgröße). Mit diesen Befehlen kann dieses immer wiederkehrende Problem der Dateiverwaltung ohne größeren Aufwand gelöst werden.

Grafik & Sound

Die Standart-Grafikbefehle wie POINT, LINE, ELLIPSE, AREA und TEXT sind erwartungsgemäß vorhanden. Dazu kommen leistungsfähige Befehle, die helfen, Problemstellungen eleganter zu lösen. Auch die Verwaltung mehrerer Bildschirmfenster (WINDOWS) ist einfach gelöst. Der wichtigste Befehl ist SET WINDOW, denn damit kann der Wertebereich für die Grafikausgabe bestimmt werden. Dann ist bei Grafikausgaben kein Umrechnen der Werte auf die Grafikpunkte des Bildschirms mehr notwendig, denn diese Arbeit übernimmt das Basic.

Auch an anderer Stelle zeigt sich TRUE BASIC von seiner besten Seite. Grafiken werden auf alle Auflösungen des ST automatisch angepaßt. Dies betrifft auch die Farben, die beim monochrom Monitor in Graustufen umgewandelt werden. Eine interessante Möglichkeit ist, daß man die Größe des Ausgabefensters frei bestimmen kann und die Grafikausgabe dann auf die Größe diese Fensters beschränkt wird.

Recht interessant ist der BOX-Befehl, der einen bestimmten Bildschirmabschnitt als BOX definiert und in einem String abspeichert. Diese BOX kann beliebig oft an jede Stelle des Bildschirms kopiert werden. Benutzt man dazu noch einen speziellen Befehl zum Löschen der Box, kann hiermit ein Objekt bewegt werden.

Wer sich gerne mit Musik beschäftigt, für den sind die Befehle PLAY und SOUND von besonderem Interesse. Mit PLAY werden die Melodien in der Standart Western Notation eingegeben. SOUND bietet dagegen eine bessere Kontrolle über die Töne und dient zur Erzeugung spezieller Geräusche. Eine Besonderheit des SOUND-Befehls ist, daß eine spezielle Option erlaubt, die Melodie im Hintergrund abzuspielen, während das Programm weiterläuft. Damit ist es leicht möglich, z. B. Spiele mit einer Unterlegmusik zu versehen.

```
do while a=0
   call clock
loop
end


MODULE timer
      SHARE start_time, #1, #2
      OPEN #1: screen 0,.8,0,1
      OPEN #2: screen .85,1,.9,1
      LET start_time = time
      CALL clock

      SUB clock
            LET t=time
            WINDOW #2
            CLEAR
            PRINT round(t-start_time,1)
            WINDOW #1
      END SUB

END MODULE
```

MODULE-Aufruf in True Basic

COMPILER UND GESCHWINDIGKEIT

Bei jedem Starten des Programms mit RUN wird der Source-Code auf Syntax-Fehler untersucht, erst wenn ein Programm in dieser Beziehung fehlerfrei ist, wird es gestartet. Dabei wird ein sogenannter Zwischencode erzeugt, der kompakt und schnell ist. TRUE BASIC ist somit kein richtiger Compiler, obwohl dieser Eindruck durch die Option 'COMPILE' aufkommen könnte. Es ist vielmehr, wie auch OMIKRON- oder GFA-Basic, ein optimierender Interpreter. Mit 'COMPILE' wird lediglich ein File erzeugt, das nicht mehr geändert werden kann und bei dem dadurch die Syntaxkontrolle und Zwischencodeerzeugung entfällt. Es gibt auch keinen RUN-TIME-Interpreter, der es erlaubt, Programme so aufzubereiten, daß sie auch Personen verwenden können, die diesen Interpreter nicht besitzen. Ein solches RUN-TIME-PACKAGE muß extra gekauft werden, der Preis liegt bei 248 DM. Damit erhält man dann aber auch die Lizenz um selbsterstellte Programme

verkaufen zu dürfen. Im Klartext bedeutet das, daß man selbsterstellte Programme, ohne dieses Programmpaket, nicht weitergeben kann. Dies ist eine erhebliche Einschränkung für den Benutzer, die man nicht unterschätzen sollte. Die Durchführung der Benchmark Tests ergab einige Schwierigkeiten. Obwohl laut Handbuch immer Variablen im Fließkommaformat verwendet werden, ergab sich bei den Schleifen ein Unterschied, wenn sie anstelle von 1 bis 1000 lief, als von 1.1 bis 1000.1. Bei der Integerschleife waren die Zeiten deutlich besser, was darauf schließen läßt, das TRUE BASIC intern auf Integeroperationen umschaltet, wenn dies möglich ist. Nicht zuletzt wegen dieser Tatsache wurde auf eine direkte Gegenüberstellung der Werte verzichtet. Bewertend kann aber gesagt werden, daß TRUE BASIC-Programme etwas langsamer ablaufen, als mit OMIKRON- oder GFA-Basic. Die Zeiten liegen allerdings dicht beieinander.

Was ST-Anwender vermissen werden

Der Grundwortschatz von TRUE BASIC nutzt viele ST-spezifischen Befehle nicht ausreichend oder gar nicht. Dies liegt daran, daß man sich an den Möglichkeiten des schwächsten Glieds, in diesem Fall dem IBM PC, orientieren mußte, um eine übertragbare Sprache gewährleisten zu können. Deshalb fehlen Befehle für 'echte' Windows, Drop-down-Menüs und vor allem die komfortablen Interruptbefehle (ON TIMER/MOUSE/MENU GOSUB). Wer diese Befehle verwenden will, der muß sich die Prozeduren entweder selbst schreiben, oder er kauft sich das 'ST Developer's Toolkit' (128 DM), in dem alle diese Funktionen als Bibliotheken abgelegt sind.

Übertragbar?

TRUE BASIC ist nicht kompatibel zum 'Standard-Basic' von Microsoft. Dies betrifft sowohl die Syntax, als auch einige Befehle, die entweder einen anderen Namen haben, oder in dieser Form gar nicht mehr existieren (z.B. LEFT$, RIGHT$, MID$). Dies bringt einige Probleme mit sich. Im Falle des IBM-PC und dessen BASICA hat man deshalb einen Converter entwickelt, der diese Programme weitgehend an TRUE BASIC anpaßt. Doch auch treten Schwierigkeiten auf, so daß einige Befehle von Hand angepaßt werden müssen. Auch beim ATARI ST gibt es Probleme beim Übertragen von OMIKRON-, GFA- und anderen Basicprogrammen. Dies betrifft an erster Stelle alle Systembefehle, die in einigen dieser Interpreter schon vordefiniert sind, aber auch eine Vielzahl von anderen Befehlen muß speziell abgeändert werden. Dabei ist auch interessant, daß TRUE BASIC nur einen numerischen Variablentyp unterstützt. Dieser arbeitet mit einer 14-stelligen Genauigkeit und entspricht dem doppeltgenauen Gleitkommaformat anderer Systeme. Diese Vereinfachung hat Vor- und Nachteile. Auf der einen Seite muß man sich keine Gedanken um den geeigneten Variablentyp machen, auf der anderen Seite kann die Genauigkeit und die Geschwindigkeit darunter leiden.

Alle Programme, die mit anderen Basic-Versionen erstellt wurden, lassen sich auf TRUE BASIC anpassen . Die systemnahen Befehle des BIOS, XBIOS und GEMDOS sind komplett über Bibliotheken ansprechbar. Die Anpassungsarbeiten sind also in erster Linie eine Frage des Aufwands oder des Preises.

Für wen eignet sich TRUE BASIC

Vergleicht man TRUE BASIC mit dem Standard, dann ist die Befehlsvielfalt überwältigend. Viele Bereiche, vor allem jedoch die Grafik, wurden um viele Befehle erweitert. Besonders interessant sind jedoch die Strukturbefehle (WHILE,UNTIL,CASE) und die Unterprogrammtechnik (DEF FUNC, SUB, LIBRARY), die eine sehr gute Strukturierung der Programme und ein Aufteilen in 'handliche' Module erlauben. Ein kleiner Nachteil ist, daß manche Basicbefehle nicht mehr kompatibel zum Sprachstandard 'MS-Basic' sind. Dadurch ergeben sich für Basic-Umsteiger einige Probleme. Auch ist ein Übertragen von Programmen problematisch, wenn auch nicht unmöglich. Ein weiterer Nachteil ist, daß man sich bei manchen Bereichen an den Möglichkeiten des IBM-PC orientieren mußte, denn dadurch werden die Fähigkeiten der anderen Systeme mit ihren grafischen Benutzeroberflächen (GEM, Intuition usw.) nicht vom Sprachkern unterstützt. Eine akzeptable Lösung sind die systemspezifischen Libraries, und die Entwickler-Toolkits, die eine komfortabel Programmierung der Benutzeroberflächen erlauben. Ist man von der Qualtität des TRUE BASIC Systems überzeugt, dann stellt sich die Frage nach den Preis. Mit 248 DM gehört es zu den teuersten Systemen, zumal dabei noch nicht die Möglichkeit zum Erstellen von unabhängigen Programmen und dem Verkauf der selbsterstellten Programmen gegeben ist. Außerdem ist es so, daß dieses RUNTIME-Pagage keinen echten Compiler beinhaltet, der eine weitere Geschwindigkeitssteigerung ermöglicht. Die Dokumentation ist sehr umfangreich: Ein Referenz-Handbuch, das alle Befehle erklärt und ein Benutzer-Handbuch, das die Besonderheiten des ST erläutert. Leider sind diese Bücher momentan nur in englischer Sprache erhältlich, was jedoch einen Computeranwender nicht erschrecken dürfte.

Als Fazit bleibt zu sagen, daß TRUE BASIC sicherlich den zur Zeit besten und ausgereiftesten Sprachschatz bietet. Außerdem gewährleisten vorhandene und geplante LIBRARIES einen ständigen Ausbau des Systems. Es ist daher auch für professionelle Anwendung gut geeignet. Diese Anwendergruppe erklärt dann den, für den privaten Anwender, recht hohen Preis des Basic-Systems.

(mn)

- \+ Strukturbefehle (DO WHILE, DO UNTIL, IF THEN ELSE)
- \+ Modultechnik (Libraries)
- \+ komfortabler Editor
- \+ viele Erweiterungen (Libraries)
- \+ portabler Code (PC, AMIGA, MAC u. a.)
- \+ Zugriff auf Systembefehle des ST
- – relativ hoher Preis
- – Run-Only-Modul nur optional
- – Handbuch nur in Englisch

Hardwarevoraussetzung:
ROM-TOS oder 1 MB

Als Erweiterungen sind erhältlich:
ST Developer's Toolkit
Advanced String Library
3-D Graphics
Mathematician's Toolkit
Sorting & Searching
je DM 128.-

Bezugsquelle:
Pfotenhauer
Microcomputer-Anwendungen
Postfach 12 67
7590 Achern
Tel.: 0 78 41 / 50 56

ST-Ecke

Wer seinen ST liebt, der schiebt...

Diesen Monat wollen wir uns im Rahmen der ST-Ecke mit zwei Themen beschäftigen. Zum einen möchte ich ein wenig näher auf die für die RS-232-Programmierung benötigten Parameter eingehen, zum anderen nehme ich mich den Schiebereglern an, die bei einigen unserer Leser Schwierigkeiten zu bereiten scheinen.

Bild 1: So könnte ein Slider aussehen

Zunächst wollen wir uns mit der vielgeplagten RS-232-C beschäftigen. Viele unserer Leser haben mit dieser Schnittstelle einigen Ärger, was sicherlich nicht nur an der verpatzten Hardware, sondern auch an der etwas spärlichen Dokumentation liegt. Deshalb möchte ich die für die Programmierung notwendige Routine RSCONF() mit ihren Parametern einmal im ganzen darlegen.

Interessant ist übrigens, daß im neuen TOS des MEGA-ST's die Treiber-Routinen völlig überarbeitet worden sind. Also sollten Sie der Meinung sein, daß bei Ihrer Schnittstelle etwas nicht stimmt, so versuchen Sie das gleiche Programm einmal mit dem neuen TOS, sofern Sie darauf Zugriff haben.

Hier nun die Routine des Xbios für C,

Rsconf (baud, ctrl, ucr, rsr, tsr, scr),
int baud, ctrl, ucr, rsr, tsr, scr;

deren Parameter allerdings Fragen aufwerfen. Dieser Aufruf ermöglicht es, die Register des USART (Universal Synchronous/Asynchronous Receiver/Transmitter) des MFP 68901 (Multi Function Peripheral-Bausteins) im Atari ST zu beschreiben und damit die serielle Schnittstelle in ihrem Verhalten zu programmieren. Es folgt die ausführliche Schreibweise der Kürzel, wobei Tx die Datenleitung des Senders und Rx die des Empfängers ist:

baud:
bestimmt die Baudrate der Übertragung

baud	Baudrate
0	19200
1	9600
2	4800
3	3600
4	2400
5	2000
6	1800
7	1200
8	600
9	300
10	200
11	150
12	134
13	110
14	75
15	50

contrl:
gibt die Art des Handshakes an

0: keinerlei Handshake (Standardeinstellung)
1: Xon/Xoff (^S /^Q)
2: RTS/CTS
3: RTS/CTS und Xon/Xoff

ucr:
USART - Kontrollregister (jede Stelle entspricht einem Bit)

.....00. keine Parität
.....10. ungerade Parität
.....11 gerade Parität
0..00... synchron, Frequenz von TC und RC ist ungeteilt
1..00... synchron, Frequenz an TC und RC durch 16 geteilt
...01... 1 Stoppbit
...10... 1.5 Stopbits
...11... 2 Stopbits
.00..... 8 Datenbits
.01..... 7 Datenbits
.10..... 6 Datenbits
.11..... 5 Datenbits

rsr:
Receiver(Empfänger)-Status-Register

gelöschtes Bit: Funktion ausschalten
gesetztes Bit: Funktion einschalten

Bit 0: Rx (Empfänger) ein- und ausschalten
Bit 1: Synchronisationszeichen (SCR – siehe weiter unten) in Puffer schreiben
Bit 2: **Synchron:**
Sync-Zeichen erkannt
Asynchron:
Startbit erkannt (Stopbit löscht dieses Bit wieder)

Bit 3: **Synchron:**
empfangenes Zeichen gleich
Asynchron:
BREAK empfangen
Bit 4: FRAME ERROR Byte < > 0 mit Stopbit = 0 empfangen
Bit 5: Paritätsfehler
Bit 6: 0= Rx-Statusregister gelesen
1= Zeichen empfangen und Rx Puffer voll
Bit 7: 0= Rx-Zeichen aus Puffer **gelesen**
1= RX-Zeichen in Puffer **geschrieben** (Buffer voll)

tsr:
Transmitter(Sender)-Status-Register

gelöschtes Bit: Funktion ausschalten
gesetztes Bit: Funktion einschalten

Bit 0: Tx einschalten
Bit 1,2: wenn Bit 0 = 0 dann
00= Tx ist hochohmig
01= Tx ist low
10= Tx = Rx (Loopback)
11= normales Tx
Bit 3: 1= Break senden, falls asynchron Übertragen wird.
0= normales Tx
Bit 4: 1= Tx nach letztem Zeichen ausschalten
0= Tx einschalten
Bit 5: Rx einschalten, wenn nach dem letzten Zeichen der Sender ausgeschaltet wurde
Bit 6: Zeichen gesendet (und damit Buffer leer)
Bit 7: Tx-Zeichen zum Datenpuffer übertragen

scr:
Synchronous Character Register
Bei der synchronen Datenübertragung wird in dieses Register das Zeichen geschrieben, das den Anfang der Datenübertragung kennzeichnet. Das heißt, daß der Rechner (bzw. der USART) zunächst wartet, bis dieses Zeichen gesendet wird und dann erst die folgenden Zeichen weitergibt.

Sicherlich ist diese Informationsmenge für jemanden, der wenig Ahnung von der RS-232-C-Schnittstelle hat recht groß. Die obige Auflistung ist nur für diejenigen unter Ihnen gedacht, die sich schon mit dieser seriellen Schnittstelle befaßt haben. Einen ausführlichen Bericht über die RS-232-C (mit Belegung der Anschlußpins sowie Tips bei der Fehlersuche bei nicht funktionierender Datenübertragung) folgt in Kürze, da die Behandlung dieses Themas den Rahmen der ST-Ecke sprengen würde.

```
  1:    #include <obdefs.h>
  2:
  3:
  4:    #define HORIZONTAL 0
  5:    #define VERTIKAL 1
  6:    #define sel_obj(tree,which) do_objc(tree,which,SELECTED)
  7:    #define desel_obj(tree,which) undo_objc(tree,which,SELECTED)
  8:
  9:    #define SLIDEBOX 0
 10:    #define VATER 4
 11:    #define SCHIEBER 6
 12:    #define SCH_W 14
 13:    #define ECHT_W 15
 14:    #define OK 12
 15:    #define WERTE 13
 16:
 17:    char *string[] = {
 18:    "ST_ECKE",
 19:    "ST - COMPUTER",
 20:    "",
 21:    "0........2........4........6........",
 22:    "B........",
 23:    "Schieberwert: ",
 24:    "Echter Wert: ",
 25:    " OK ",
 26:    "0000",
 27:    "000"};
 28:
 29:
 30:    int  IMAGO[] = {
 31:    0x180, 0x180, 0x3C0, 0x3C0,
 32:    0x7E0, 0x7E0, 0xFF0, 0xFF0,
 33:    0x1FF8, 0x1FF8, 0x1FF8, 0x3FFC,
 34:    0x3FFC, 0x7FFE, 0x7FFE, 0xFFFF,
 35:    0xFFFF, 0x0, 0x0, 0xFFFF,
 36:    0xFFFF, 0x7FFE, 0x7FFE, 0x3FFC,
 37:    0x3FFC, 0x1FF8, 0x1FF8, 0x1FF8,
 38:    0xFF0, 0xFF0, 0x7E0, 0x7E0,
 39:    0x3C0, 0x3C0, 0x180, 0x180,
 40:    0x0, 0x0, 0x0, 0x0,
 41:    0x0, 0x0, 0x0, 0x0,
 42:    0x0, 0x0, 0x0, 0x0};
 43:
 44:    BITBLK bit[] = {
 45:    &IMAGO[0], 2, 48, 0, 0, 1};
 46:
 47:    TEDINFO ted[] = {
 48:    string[1], string[2], string[2], 5, 6, 2, 0x1180, 0x0, -1, 14,1,
 49:    string[3], string[2], string[2], 5, 6, 0, 0x1180, 0x0, -1, 38,1,
 50:    string[4], string[2], string[2], 5, 6, 0, 0x1180, 0x0, -1, 10,1,
 51:    string[5], string[2], string[2], 3, 6, 0, 0x1180, 0x0, -1, 15,1,
 52:    string[6], string[2], string[2], 3, 6, 0, 0x1180, 0x0, -1, 14,1,
 53:    string[8], string[2], string[2], 3, 6, 1, 0x1180, 0x0, -1, 5,1,
 54:    string[9], string[2], string[2], 3, 6, 1, 0x1180, 0x0, -1, 4,1};
 55:
 56:    OBJECT baum[] = {
 57:    -1, 1, 1, G_BOX, NONE, OUTLINED, 0x21100L, 80,94,471,231,
 58:    0, 2, 13, G_BOX, NONE, SHADOWED, 0xFD1121L,40,32,376,160,
 59:    3, -1, -1, G_STRING, NONE, NORMAL, string[0], 152,8,56,16,
 60:    4, -1, -1, G_TEXT, NONE, NORMAL, &ted[0], 144,22,78,16,
 61:    7, 5, 6, G_BOX, NONE, NORMAL, 0xFD1120L, 49,64,281,36,
 62:    6, -1, -1, G_BOX, NONE, NORMAL, 0xFF1141L, 0,15,281,6,
 63:    4, -1, -1, G_IMAGE, TOUCHEXIT, NORMAL, &bit[0], 0,0,17,36,
 64:    8, -1, -1, G_TEXT, NONE, NORMAL, &ted[1], 53,52,222,6,
 65:    9, -1, -1, G_TEXT, NONE, NORMAL, &ted[2], 268,52,54,6,
 66:    12, 10, 11, G_BOX, NONE, NORMAL, 0xFF1100L, 72,109,238,41,
 67:    11, -1, -1, G_TEXT, NONE, NORMAL, &ted[3], 6,3,112,16,
 68:    9, -1, -1, G_TEXT, NONE, NORMAL, &ted[4], 6,22,104,16,
 69:    13, -1, -1, G_BUTTON, 0x7, NORMAL, string[7], 329,6,37,33,
 70:    1, 14, 15, G_IBOX, NONE, NORMAL, 0x10100L. 192,111,92,39,
 71:    15, -1, -1, G_BOXTEXT, NONE, NORMAL, &ted[5], 3,1,64,16,
 72:    13, -1, -1, G_BOXTEXT, LASTOB, NORMAL, &ted[6], 3,21,64,16};
 73:
 74:
 75:
 76:    long tree;                  /* Adresse der Baumstruktur */
 77:    GRECT schieber, vater;      /* Boxkoordinatenstruktur */
 78:    int pos_alt, exit_obj=VATER, position, neu_pos;
 79:    char *ptadr();              /* Routine gibt nicht Integer zurück */
 80:
 81:    main()
 82:    {
 83:       int i;
 84:       OBJECT *adr;
 85:
 86:       appl_init();                     /* Applikation anmelden */
 87:
 88:       tree = (long)&baum[0];
 89:
 90:       box_draw(tree,320,200,40,40);          /* Box zeichnen */
 91:
 92:       objc_xywh(tree,SCHIEBER,&schieber);    /* Koordinaten des Schiebers */
 93:       objc_xywh(tree,VATER    ,&vater   );   /* Koordinaten des Vaters */
 94:
 95:       pos_alt = vater.g_w - schieber.g_w;    /* Beide Breiten subtrahieren */
 96:
 97:       while (exit_obj!=OK)                   /* Bis OK gedrückt */
 98:       {
 99:          exit_obj = form_do(tree,0);
100:
101:          /* Bei Übergabe von "VATER" und "SOHN" und der Schieberichtung,
```

Nun noch ein kleiner Wehrmutstropfen: Der Grund für das Nicht-Zustande-Kommen einer seriellen Übertragung könnte das Fehlen der genauen Parameter sein, die Sie oben im Betriebssytemaufruf finden. Dazu läßt sich folgendes sagen: Es ist bekannt, daß weder der Software-Handshake Xon/Xoff noch der Hardware-Handshake RTS/CTS beim Atari funktionieren!!! Da aber viele Geräte mit diesen Handshakes arbeiten, kommen keine vernünftigen Übertragungen zustande. Dieser Fehler liegt im Betriebssystem des Atari und kommt bei Benutzung der Betriebssystemroutinen zum Tragen. Es steht natürlich jedem frei seine eigenen Übertragungsroutinen zu schreiben. **Ein kleiner Trost (wie oben schon erwähnt):** Im neuesten TOS der MEGA-ST's, daß übrigens auch auf den alten Rechner läuft, sind die Routinen völlig neu geschrieben worden – es bleibt zu hoffen, daß dieser gravierende Fehler nun behoben worden ist.

Die Slider

Als zweites Thema möchte ich mich mit Schiebereglern befassen. Diese Schieberegler, auch neudeutsch 'Slider' genannt, tauchen in vielen Programmen und Fenstern auf und ermöglichen es zum Beispiel auf einfache Weise, kontinuierliche Werte einzustellen.

Um mit diesen Slidern arbeiten zu können, sollte man sich recht gut mit der Objektstruktur von Resourcen (ST Computer Sonderheft Nr.2) auskennen. Was sind also Slider ? Prinzipiell ist ein Slider nichts anderes als ein Objekt, das innerhalb eines Objektes verschoben werden kann. Um sich diese Tatsache zu verdeutlichen, schauen Sie sich bitte Bild 1 an, in dem ein Schieberegler dargestellt ist. Das verschiebbare Objekt, hier interessanterweise keine BOX, sondern ein ICON mit dem Namen Schieber, befindet sich in dem Objekt VATER. Wichtig ist, daß dies objektorientiert gemeint ist, Schieber muß tatsächlich in der Objektstruktur eine Ebene unter dem Objekt Vater stehen. Dieses Vaterobjekt begrenzt den Raum, in dem der Schieber verschoben werden kann. Will man einen horizontalen Schieber realisieren, wie hier gezeigt wird, so sollte man die Höhe des Vaterobjektes genauso hoch anlegen wie die des Schiebers – analog dazu, wird die Breite eines vertikalen Schiebers angepaßt.

```
102:            sorgt die Funktion graf_slidebox dafür, daß eine BOX in Größe
103:            des Sohns innerhalb des Vaters in Schieberichtung verschoben
104:            werden kann. Rückgabe ist Wert zwischen 0 und 1000 */
105:
106:         position = graf_slidebox(tree,VATER,SCHIEBER,HORIZONTAL);
107:         sprintf(ptadr(tree,SCH_W,0),"%04d", position);  /* Zahl ausgeben */
108:         neu_pos = (int)(((long)pos_alt*(long)position)/1000L); /* umrechnen */
109:         sprintf(ptadr(tree,ECHT_W,0),"%3d", neu_pos);   /* Zahl ausgeben */
110:         ((OBJECT*)(tree+24*SCHIEBER))->ob_x = neu_pos;  /* Zahl einschreiben */
111:         objc_update(tree,WERTE,MAX_DEPTH);              /* Werte darstellen */
112:         objc_update(tree,VATER,MAX_DEPTH);              /* Schieber neuzeichnen */

113:      }
114:      box_undraw(tree,320,200,40,40);           /* Box löschen */
115:      desel_obj(tree,exit_obj);                 /* OK-Knopf deselektieren */
116:
117:      rsrc_free();                              /* Speicher freigeben */
118:
119:      appl_exit();                              /* Applikation abmelden */
120:
121:   }
122:
123:   box_draw(tree,x,y,w,h)
124:   int x, y, w, h;
125:   long tree;
126:   {
127:        int xdial,ydial,wdial,hdial;
128:
129:        form_center ( tree, &xdial, &ydial, &wdial, &hdial );
130:        form_dial ( 0, x, y, w, h, xdial, ydial, wdial, hdial );
131:        form_dial ( 1, x, y, w, h, xdial, ydial, wdial, hdial );
132:        objc_draw ( tree, 0, 8, xdial, ydial, wdial, hdial );
133:   }
134:
135:   box_undraw(tree,x,y,w,h)
136:   int x, y, w, h;
137:   long tree;
138:   {
139:        int xdial,ydial,wdial,hdial;
140:
141:        form_center ( tree, &xdial, &ydial, &wdial, &hdial );
142:        form_dial ( 2, x, y, w, h, xdial, ydial, wdial, hdial );
143:        form_dial ( 3, x, y, w, h, xdial, ydial, wdial, hdial );
144:   }
145:
146:   char *ptadr(tree,ind,clear)
147:   long tree;
148:   int ind;
149:   {
150:       OBJECT *obj;
151:       TEDINFO *ted;
152:       char *text_adr;
153:
154:       obj=(OBJECT*)(tree+241*ind);
155:       ted=(TEDINFO*)(obj->ob_spec);
156:       text_adr=ted->te_ptext;
157:       if (clear)
158:          *text_adr=0;
159:       return(text_adr);
160:   }
161:
162:   undo_objc(tree,which,bit)
163:   long tree;
164:   int which,bit;
165:   {
166:     OBJECT *zeiger;
167:
168:
169:      zeiger = (OBJECT*) (tree+ which*sizeof(OBJECT));
170:      zeiger->ob_state &= ~bit;
171:   }
172:
173:   do_objc(tree,which,bit)
174:   long tree;
175:   int which,bit;
176:   {
177:     OBJECT *zeiger;
178:
179:      zeiger = (OBJECT*) (tree+ which*sizeof(OBJECT));
180:      zeiger->ob_state |= bit;
181:   }
182:
183:   objc_xywh(tree, obj, p)
184:   long tree;
185:   int obj;
186:   GRECT *p;
187:   {
188:      objc_offset(tree,obj,&p -> g_x, &p -> g_y);
189:      p -> g_w = ((int) *((int *)(tree+24*obj+20)));
190:      p -> g_h = ((int) *((int *)(tree+24*obj+22)));
191:   }
192:
193:   objc_update(tree, obj, depth)
194:   long tree;
195:   int obj, depth;
196:   {
197:      GRECT box;
198:
199:      objc_xywh(tree,obj,&box);
200:      objc_draw(tree,obj,depth,box.g_x,box.g_y,box.g_w,box.g_h);
201:   }
```

Listing 1: Das Listing zu Bild 1

Wird nun ein Objekt angeklickt, das einen Schieber darstellt, so springt man in die Routine graf_slidebox, die wie folgt definiert ist:

slidenummer = graf_slidebox(baum, vater, schieber, richtung)

Baum: Dies ist die Anfangsadresse der Baumstruktur

Vater: Objektindex des Vaterobjektes

Schieber: Objektindex des Schiebers

Richtung: Gibt die gewünschte Schieberichtung an.
HORIZONTAL = 0
VERTIKAL = 1

Slidenummer: enthält eine die Position des Schiebers.

Dabei übergibt man, wie zu erkennen ist, die Baumadresse und die betroffenen Objekte, sowie die Schieberichtung. Um den Rückgabewert zu verstehen, müssen Sie sich die Breite Ihres Schiebebereichs (im Prinzip die Breite des Vaterobjektes – weiteres siehe unten) in 1000 Schritte eingeteilt denken. Ist der Schiebebereich nur 250 Pixel groß, so würde eine Verschiebung des Sliders um ein Pixel eine Erhöhung des Rückgabewertes um vier bedeuten. Nun zum Schiebebereich: Wenn beispielsweise das Vaterobjekt eine Breite von 100 und der Schieber eine Breite von 20 hat, so bedeutet dies für den Schiebebereich eine Länge von 80 Pixeln, denn der Schieber schränkt durch seine eigene Breite die Verschiebung ein, da er das Vaterobjekt nicht verlassen darf. Diese Korrektur findet man im Programm in der Anweisung pos_alt = vater.g_w – schieber.g_w.

Leider wird bei der ganzen Aktion nicht der eigentliche Schieber, sondern nur dessen Boxumriß verschoben, so daß wir nach dem Loslassen den Schieber an die richtige Stelle setzen müssen. Dazu errechnen wir uns die Koordinate, indem wir den Schiebebereich (pos_alt) durch 1000 teilen und mit dem Schieberwert multiplizieren (Dreisatz):

$$\text{Objektposition} = \frac{\text{Schiebebereich}}{1000} * \text{Schieberwert}$$

Diesen Wert schreiben wir in die Objektstruktur des Schiebers und zeichnen ihn neu. Erst jetzt steht der Schieber an der Stelle, an der sein Umriß losgelassen wurde.

Zum Schluß möchte ich noch auf ein paar Dinge eingehen, die mit Slidern möglich sind. Schöne Beispiele finden Sie im Kontrollfeld des Atari ST: Wie Sie sicherlich erkannt haben, kann man nicht nur (langweilige) Boxen als Schieber verwenden, sondern auch ICONS oder IMAGES (siehe Kontrollfeld). In unserem Beispiel könnte man anstatt des IMAGES ein ICON verwenden und in den Icontext die Position des Schiebers schreiben. Theoretisch ist es sogar möglich, benutzerdefinierte Objekte einzubinden.

In einer der nächsten Ausgaben werden wir uns dann auch mit einer eigens geschriebenen Routine beschäftigen, bei der man gleichzeitig zwei Schieber bewegen kann, oder zum Beispiel der aktuelle Schieberwert während des Verschiebens angegeben wird... Bis dann...

(SH/HE)

GFA DRAFT PLUS

GFA DRAFT PLUS ist die Weiterentwicklung von GFA DRAFT, das wir in der Februarausgabe vorgestellt haben. Es war damals eines der wenigen und eines der günstigsten CAD-Programme. Es war jedoch in seiner Bedienung teilweise recht umständlich und träge. In diesem Bericht werden deshalb vorwiegend die Verbesserungen und Erweiterungen besprochen.

Bild 1: Die wichtigsten Einstellungen für das Projekt

Linienoperationen

Der erste positive Punkt bei GFA DRAFT PLUS ist, daß es nur noch eine Menüzeile gibt und die Einstellungen jetzt in umfangreicheren Dialogboxen vorgenommen werden können. Damit ist es endlich möglich, Einstellungen auf eine akzeptable Art vorzunehmen. Nach dem Starten des Systems kann sofort mit dem Ziehen von Linien begonnen werden. Die Linienattribute (Typ, Breite, Art), werden in einem einzigen Menü eingestellt (das war nicht immer so einfach!). Allerdings zeigen sich schon hier erste Mängel. So beginnt die Linienbreite erst bei 0.3 mm, obwohl man bei jeder technischen Zeichnung eine Linienbreite von 0,25 mm benötigt. Als zweites wundert man sich über den Sinn der Linienarten. Diese können zwar eingestellt und benutzt werden, allerdings können oft benötigte Linieneinstellungen nicht abgespeichert werden. Laut Handbuch ist der Sinn dieser Option das spätere Trennen verschiedener Linien z. B. nach elektrischen und hydraulischen Leitungen.

Die Funktionen, die sich auf Linien beziehen sind zahlreich und einfach zu handhaben:

- Linien knacken (teilt Linien an Schnittpunkten)
- Linien trimmen (schneidet überstehende Linienteile ab)
- Linie 2 zu 1 (faßt zwei Linien zu einer zusammen)
- Linienlänge ändern
- Linien einrasten
- Parallele zu einer Linie
- Tangente zu einem Kreis
- Lot auf Gerade
- Winkel zu Gerade

Die Funktionen Parallele und Tangente sind neu hinzugekommen und bedürfen wohl keiner Erklärung. Sehr gut ist die Funktion Linie einrasten. Dabei werden Linien, die sich in einem bestimmten Bereich (einstellbarer Fangradius) befinden, miteinander verbunden. Diese Funktion ist besonders zum Erstellen von geschlossenen Flächen wichtig, wenn diese schraffiert werden sollen.

Weitere Zeichenfunktionen

Neben Linien-Funktionen bietet GFA DRAFT PLUS natürlich auch noch andere Optionen:

- Rechteck
- Kreis(segment), Kreis durch drei Punkte
- Ellipsen(segment)
- Zeichenstz (verschiedene Größen, spiegeln, drehen um 90)

Fenstermanipulationen

Wenn Elemente und Objekte schon erstellt sind, dann ist es immer noch möglich, eine Fülle von Manipulationen durchzuführen. Diese beziehen sich entweder nur auf ein selektiertes Objekt oder aber auf einen, mit der Fensterfunktion, festgelegten Bereich. Manipulationen sind:

- Kopieren
- Verschieben
- Löschen
- Zoomen
- Drehen
- Spiegeln
- Dehnen/Stauchen
- Verzerren

Schraffieren und Bemaßen

Diese Funktionen sind beim Anfertigen von technischen Zeichnungen sehr wichtig. Zum Schraffieren ist im allge-

meinen ein geschlossener Streckenzug notwendig, doch bei GFA DRAFT PLUS kann auch der Bereich zwischen zwei Linien mit einer Schraffur versehen werden. Auch das Bemaßen von Strecken ist recht einfach. Zuerst werden die Maßhilfslinien gezogen, die sich ziemlich problemlos auf Linien einklinken. Zwischen diesen Hilfslinien wird dann der Bemaßungspfeil gezogen und mit einem weiteren Mausklick erscheint die Maßzahl. Dieses Verfahren ist zwar nicht das komfortabelste, aber es ist im Vergleich zur Handarbeit eine große Hilfe.

Neu hinzugekommen ist eine Funktion, die bei der Beschriftung von z. B. Widerständen automatisch die Nummer erhöht (R11, R12 usw.). oder eine Bezeichnung beibehält.

Bild 2: Das Druckmenü

Löschen

Das Löschen ist schon bei GfA DRAFT sehr einfach gewesen. Mit BACKSPACE wird die jeweils letzte Eingabe gelöscht. Dies kann so lange fortgeführt werden, bis das Blatt wieder leer ist. DELETE löscht dagegen das Objekt, das dem Cursor am nähesten ist. Es kann aber auch ein markierter Bereich gelöscht werden. Eine neue Option erlaubt außerdem das Löschen von Objekten. Bei GFA DRAFT konnte nämlich ein Kreis nur durch Löschen einer Unzahl von Vektoren entfernt werden. Bei aktiviertem Objekt löschen geht dies auf einen Schlag.

Symbole

Oft benötigte Linienkombinationen oder ähnliches können als ein Symbol abgespeichert werden. Beim Einladen muß man dann aber immer über die Fileselektorbox eine Auswahl treffen. Deshalb ist meistens eine Funktionstastenbelegung sinnvoll. Das Objekt wird dann zwar immer noch von Diskette nachgeladen (obwohl doch der ST so viel Speicher hat!), aber die Objektauswahl entfällt. Diese Möglichkeit ist besonders für Platinenlayouts interessant, wenn z. B. eine Vielzahl Widerstände benötigt werden.

Ebenfalls für diese Anwendung geeignet, ist die Option, mehrere Zeichnungen als Ebenen übereinander darzustellen.

Kommandosprache

Eine weitere neue Option ist die Kommandosprache, die nun implementiert ist. Damit können Zeichnungen mit Befehlsfolgen erstellt werden, was natürlich eine größere Genauigkeit zur Folge hat. Die Sprache ist sehr einfach und wird in ähnlicher Form auch bei anderen CAD-Systemen, u. a. Schneidbrennersteuerungen verwendet. FD 50 zeichnet z. B. eine Linie mit der Länge 50 mm. Die Sprache enthält aber auch komplexere Befehlssequenzen wie Schleifen und Makros.

Der Aufbau dieser Dateien wird im Handbuch erklärt, so daß sie auch von einem anderen Programm erstellt werden können. Denkbar wäre demnach das Berechnen von Werten in einem eigenen Programm und Ausgabe in Form der Kommandosprache des GFA DRAFT PLUS.

Drucken und Plotten

Die schönste Zeichnung nutzt nichts, wenn man sie nicht zu Papier bringen kann. Doch neben einer stattlichen Anzahl schon installierter Druckertreiber, wird auch die Anpassung jedes beliebigen Druckers/Plotters erklärt. Dem Ausdruck auf diesen Geräten ist jeweils eine Dialogbox gewidmet. Dort werden dann Blattgröße, Verschiebung, Zoomfaktor, Ausrichtung, Druckdichte und ähnliches eingestellt (siehe Bild). Außerdem ist es jetzt möglich einen bestimmten Ausschnitt auszugeben.

Bild 3: Oft benutzt: Das Linienmenü

Bild 4: Beispiel eines Arbeitsbildschirms

Wertung

Eine Wertung von GFA DRAFT PLUS ist nicht ganz einfach. Gemessen an seinem Vorgänger, bietet es eine Fülle wichtiger Zusatzfunktionen und viele gravierende Fehler und Mängel sind behoben worden. Dennoch ist die Bedienung nicht immer zufriedenstellend. Wer technische Zeichnungen anfertigt, der braucht abwechselnd dünne, gestrichelte und dickere, durchgezogene Linien. Es gibt aber keine Möglichkeit, diese Linienarten abzuspeichern, um schnell zwischen ihnen wechseln zu können. Hier, und auch an anderer Stelle, wäre es sinnvoll einige Ikons auf dem Bildschirm zu plazieren oder Funktionstasten dafür vorzusehen. Dann könnten die wichtigsten Funktionen schnell angewählt werden, ohne in Menüs oder Dialogboxen rumfahren zu müssen.

Auch das Arbeiten mit einer sichtbaren Linienbreite ist gewöhnungsbedürftig, weil man nicht die tatsächlichen Breiten kontrollieren kann. Dies ist jedoch die einzige Art, einen schnellen Bildaufbau zu erreichen, denn wenn alle Linien in ihrer tatsächlichen Breite dargestellt würden, dann benötigte GFA DRAFT PLUS dafür eine Ewigkeit. Für den Anwender bedeutet dies ein gutes Gedächtnis für die schon gezeichneten Linien, aber auch für die momentan eingestellte Breite.

GFA DRAFT PLUS kostet DM 349. Dieser Preis ist für ein CAD-Programm durchaus angemessen, allerdings gibt es für den ST noch einige andere in dieser Preisklasse. Man sollte sich deshalb genau überlegen, was für den eigenen Bedarf das Geeignetste ist. Arbeiten kann man mit GFA DRAFT PLUS allemal. Viele Funktionen und Features besitzt es und deren Leistungsfähigkeit ist sehr hoch. Was fehlt, ist in erster Linie der Komfort und die Leichtigkeit der Bedienung, die man sonst von vielen ST-Produkten gewohnt ist.

(MN)

Floppy-spielereien

Teil 4

Nachdem in der letzten Folge kräftig über Adressen und Bit's sowie Komandos und Register gesprochen wurde, wird diesmal der TRACKMON vorgestellt. Es handelt sich dabei um eine Erweiterung des MINIMON. Verwendet werden die zwei Routinen vom letztenmal, SELECT und LOCKSLEY, diesmal allerdings als Data-Zeilen.

MINIMON kräftig aufgeblasen

Im Listing des MINIMON, der in der ST Computer Juli/August vorgestellt wurde, hatte ich ja eigens eine Auswahl zwischen Sektormonitor und Trackmonitor vorgesehen. In diesem Heft finden Sie nun die Prozeduren, die Sie brauchen, um aus dem MINIMON den EXTENDED MINIMON zu machen. In dem Programm von damals müssen allerdings drei kleine Eingriffe gemacht werden:

Die Routine DUMP benötigt jetzt einen Parameter. Ändern Sie also den Prozedurkopf Procedure Dump in Procedure Dump(L) ab. Weiterhin ändern Sie in der siebten Zeile die For-Anweisung in For T=1 To L Step 16 ab. Außerdem müssen Sie in der Procedure Sektormon noch etwas drehen: Statt If M$="D" @ Dump Endif, muß es heißen If M$="D" @ Dump(512) Endif (natürlich müssen die einzelnen Anweisungen in getrennte Zeilen!).

Wenn Sie statt der Datazeilen lieber die passenden Assemblerprogramme abtippen wollen, können Sie sich auch die Zahlenwüste sparen und statt dessen das assemblierte LOCKSLEY.PRG mit dem ebenfalls abgedruckten PRG-

```
'
' Minimon für den ST
' Written 1987 by Claus Brod
'                 Am Felsenkeller 2
'                 8772 Marktheidenfeld
'
Cls
Do
  Print
  Print "Minimon ST - (C) 1987 by Claus Brod"
  Print
  Repeat
    Print "Sektormonitor oder Trackmonitor oder Quit (S/T/Q)?"
    A$=Upper$(Input$(1))
  Until A$="S" Or A$="T" Or A$="Q"
  Exit If A$="Q"
  If A$="S"
    @Sektormon
  Else
    @Trackmon
  Endif
Loop
End
'
'
Procedure Gibmirzeit
  Alert 1,"Funktion noch nicht|implementiert.",1,"OK",A
Return
'
' Prozedur Trackmon
' initialisiert den Trackmonitor
' zeigt in einer Schleife das Menü an, fragt auf Tastendruck ab
' und verteilt auf die Unterroutinen
'
Procedure Trackmon
  @Init_trackmon
  Cls
  Prn%=0
  Richtung%=0
  Seite%=0
  Drive%=0
  Steprate%=1               ! 3ms Steprate
  R$=Chr$(27)+"p"
  O$=Chr$(27)+"q"
  @Rst                      ! auf Track 0 zurückfahren
  Do
    Print
    Print "******************** TRACKMON (C) 1987 Claus Brod ********************
    @Prntable
    Print R$;"0";O$;" Track 0, ";R$;"1";O$;" Step, ";R$;"2";O$;" Step-In, ";
    Print R$;"3";O$;" Step-Out, ";R$;"4";O$;" Seek, ";R$;"5";O$;" Rd Sec, ";
    Print R$;"6";O$;" Wr Sec"
    Print R$;"7";O$;" Rd Adr, ";R$;"8";O$;" Rd Trk, ";R$;"9";O$;" Wr Trk, ";
    Print R$;"A";O$;" IRQ, ";R$;"D";O$;"ump Buf, ";R$;"S";O$;"eite (";Seite%;
    Print "), ";R$;"P";O$;"rn (o";
    If Prn%
      Print "n)"
    Else
      Print "ff)"
    Endif
    Print R$;"R";O$;"egister, ";R$;"Q";O$;"uit"
    M$=Upper$(Input$(1))
    Exit If M$="Q"
    If M$="S"
      Seite%=1-Seite%
    Endif
    If M$="P"
      Prn%=1-Prn%
    Endif
    If M$="D"
      @Dump_buf
    Endif
    If M$="A"
      @Irq                                ! FDC unterbrechen
    Endif
    If M$="R"
      @Register
    Endif
    If M$<="9" And M$>="0"
      On Val(M$)+1 Gosub Rst,Step,Step_in,Step_out,Seek,Rdsec,Wrsec,Rdadr
      On Val(M$)-7 Gosub Rdtrk,Wrtrk
    Endif
  Loop
Return
'
'
Procedure Init_trackmon
  Print "Einen Moment, bitte!"
  Restore Locksley
  @Readprog(1000)
  ' LOCKSLEY.S einlesen
  Inter$=Prg$
  Inter=Varptr(Inter$)
  Opcode=Inter+3
  Restore Select
  @Readprog(1000)                           ! Reserve für Erweiterung
```

Listing 1

```
  ' SELECT.S einlesen
  Sel$=Prg$
  Sel=Varptr(Sel$)
  Laufwerk=Sel+1
Return
'
'
Procedure Select
  Poke Laufwerk,Seite%+2                         ! Laufwerk A, Seite 'Seite%'
  Call Sel
Return
Procedure Deselect
  Poke Laufwerk,0                                ! deselektieren
  Call Sel
Return
'
Procedure Mach_schon(O%)
  @Select                                        ! Laufwerk selektieren
  Poke Opcode,O%
  Call Inter                                     ! Kommando ausführen
  @Deselect                                      ! Laufwerk abwählen
Return
'
Procedure Register
  @Gibmirzeit                                    ! noch nicht implementiert
Return
'
' Prozedur Rst
' fährt den Lesekopf auf Spur 0 zurück
'
Procedure Rst
  Print R$;"Restore";O$
  Track%=0
  Richtung%=0
  @Mach_schon(0+Steprate%)                       ! Restore-Befehl
Return
'
' Prozedur IRQ
' unterbricht den Floppycontroller bei der Arbeit
'
Procedure Irq
  Print R$;"IRQ";O$
  @Mach_schon(208)                               ! IRQ-Befehl
Return
'
' Prozedur Step
' fährt den Lesekopf einen Schritt in die eingeschlagene Richtung
'
Procedure Step
  Print R$;"Step";O$
  Track%=Track%+Richtung%
  @Mach_schon(32+16+Steprate%)                   ! Step mit Track-Update
Return
'
' Prozedur Step_in
' fährt den Lesekopf einen Schritt nach innen
'
Procedure Step_in
  Print R$;"Step-in";O$
  Richtung%=1
  Track%=Track%+Richtung%
  @Mach_schon(64+16+Steprate%)                   ! Step-in mit Update
Return
'
' Prozedur Step-out
' fährt den Kopf einen Schritt nach außen
'
Procedure Step_out
  Richtung%=-1
  Print R$;"Step-out";O$
  Track%=Track%+Richtung%
  @Mach_schon(96+16+Steprate%)                   ! Step-out mit Update
Return
'
' Prozedur Seek
' fährt den Lesekopf auf die gewünschte Spur
'
Procedure Seek
  Print R$;"Seek";O$
  Input "Zieltrack";Trk
  If Trk>Track%
    Richtung%=1
  Else
    If Trk<Track%
      Richtung%=-1
    Endif
  Endif
  Track%=Trk
  Poke Inter+5,Trk                               ! Spurnummer abliefern
  @Mach_schon(16+Steprate%)                      ! Seek-Befehl
Return
'
' Prozedur Rdsec
' liest Sektor(en) auf aktuellem Track ein
'
Procedure Rdsec
  Print R$;"Read sector";O$
  Input "Welcher Sektor";Sek
  Poke Inter+7,Sek                               ! Sektornummer schreiben
  Input "Wieviel Bytes";Laenge
  Buf$=Space$(12*512)
  Dpoke Inter+8,Laenge                           ! Länge der Übertragung
  Lpoke Inter+10,Varptr(Buf$)                    ! Pufferadresse
  @Mach_schon(128+16)                            ! Read multiple sectors
Return
'
' Prozedur Rdadr
' Liest Adreßfelder auf aktuellem Track ein
'
Procedure Rdadr
  Print R$;"Read Adress";O$
  Input "Wieviele Adreßfelder";Laenge
  Dpoke Inter+8,Laenge                           ! Zahl der Adreßfelder
  Laenge=Laenge*6                                ! 6 Byte pro Adreßfeld
  Buf$=Space$(512)
  Lpoke Inter+10,Varptr(Buf$)                    ! Pufferadresse
  Buf2$=Space$(100)
  Lpoke Inter+14,Varptr(Buf2$)                   ! Pufferadresse für Status
  @Mach_schon(192)                               ! Rd-Address-Befehl
Return
'
' Prozedur Wrsec
' schreibt Sektor(en) auf aktuellen Track
'
Procedure Wrsec
  Print R$;"Write sector";O$
  Input "Welcher Sektor";Sek
  Poke Inter+7,Sek                               ! Sektornummer
  Input "Wieviel Bytes";Laenge
  Dpoke Inter+8,Laenge                           ! Länge der Übertragung
  Lpoke Inter+10,Varptr(Buf$)                    ! Pufferadresse
  @Mach_schon(160+16)                            ! Write multiple sectors
Return
'
' Prozedur Rdtrk
' Aktuellen Track einlesen
'
Procedure Rdtrk
  Print R$;"Read Track";O$
  Buf$=Space$(8000)
  Lpoke Inter+10,Varptr(Buf$)                    ! Pufferadresse
  Input "Wieviel Bytes";Laenge
  Dpoke Inter+8,Laenge                           ! Länge der Übertragung
  @Mach_schon(192+32)                            ! Read Track
Return
'
' Prozedur Wrtrk
' Einen Track formatieren
'
Procedure Wrtrk
  Print R$;"Write Track";O$
  T$=Space$(8000)
  ' liest die Trackdaten ab dem Label Trackdaten ein
  ' Format der Datazeilen:
  ' zu schreibendes Byte, danach Anzahl (wie oft soll dieses Byte
  ' geschrieben werden)
  ' "***,***" heißt Schluß
  Restore Trackdaten
  Cn=1
  Do
    Read B$,Z$
    @Hextodec(B$)
    B=$
    @Hextodec(Z$)
    Z=$
    Exit If B$="***"
    If Upper$(B$)="TRACK"
      B=Track%
    Endif
    If Upper$(B$)="SIDE"
      B=Seite%
    Endif
    For T=1 To Z
      Print B'
      Mid$(T$,Cn,1)=Chr$(B)
      Inc Cn
    Next T
  Loop
  Print
  Lpoke Inter+10,Varptr(T$)                      ! Pufferadresse
  Input "Wieviel Bytes";Laenge
  Dpoke Inter+8,Laenge                           ! Länge der Übertragung
  @Mach_schon(15*16)                             ! Write Track
Return
'
' Prozedur Dump_buf
' Gibt ersten oder zweiten Puffer aus
' und erlaubt ihn zu edieren
'
Procedure Dump_buf
  Repeat
    Print "Ersten oder zweiten Puffer ausgeben (1/2)?"
```

Fortsetzung von Listing 1

TOLST.BAS in die Datazeilen wandeln lassen. Die müssen Sie dann hinter das Label Locksley: 'mergen'. Ähnliches gilt für SELECT.S und das Label Select:.

Den Trackmonitor des MINIMON verstehe ich hauptsächlich als Demonstration der Floppyprogrammierung und nicht als Beispiel für strukturierte und besonders einwandfreie Programmiertechnik. Zudem können Sie Er soll zur Anregung dienen. Zudem können Sie mit dem EXTENDED MINIMON erste Experimente machen, die Sie tiefer ins Reich des Lesekopfes führen.

Wenn Sie beim Start des EXTENDED MINIMON 'T' für 'Trackmonitor' drücken, dauert es einen Moment, bis das Programm die Datazeilen eingelesen hat. Dann führt der Trackmonitor zuerst einen 'Restore'-Befehl aus, um den Lesekopf auf Track 0 und damit auf eine definierte Position zu bringen. Sie bekommen dann ein Menü samt ausführlicher Statusübersicht. In der Statusübersicht werden die Ein/Ausgabeparameter von LOCKSLEY angezeigt. Die Bedeutung der Felder hängt im allgemeinen auch vom jeweiligen Befehl ab, nachzulesen in der Anleitung zu LOCKSLEY.S. Wie der FDC- und DMA-Status decodiert wird, erfahren Sie in der nächsten Ausgabe, in der EXTENDED MINIMON um die 'Register'-Routine erweitert wird, die eine Mehrfachfunktion haben wird:

Zum einen wird man mit ihr jedem Register des Floppycontrollers auf die

```
      Ch$=Input$(1)
    Until Ch$="1" Or Ch$="2"
    If Ch$="1"
      Sec$=Buf$
      @Dump(Laenge)
    Else
      Sec$=Buf2$
      @Dump(Len(Buf2$))
    Endif
    Print R$;"Edit (Y/N)?";O$
    A$=Upper$(Input$(1))
    If A$="Y"
      If Ch$="1"
        Sec$=Buf$
        @Edit
        Buf$=Sec$
      Else
        Sec$=Buf2$
        @Edit
        Buf2$=Sec$
      Endif
    Endif
Return
'
' Prozedur Prntable
' Statusmeldung ausgeben
'
Procedure Prntable
  Inter=Varptr(Inter$)
  Opcode=Inter+3
  Sel=Varptr(Sel$)
  Laufwerk=Sel+3
  Print "Opcode                      :";Dpeek(Inter+2);Tab(40);
  If Dpeek(Inter+30)=0
    Print "Kein ";
  Endif
  Print "Timeout!"
  Print "Track                       :";Track%;Tab(40);
  Print "Sektor                      :";Dpeek(Inter+6)
  Print "Adresse des Puffers 1       :";Lpeek(Inter+10);Tab(40);
  Print "Adresse des Puffers 2       :";Lpeek(Inter+14)
  Print "FDC-Status                  :";Peek(Inter+19);Tab(40);
  Print "DMA-Status                  :";Peek(Inter+21)
  Print "Startadresse DMA            :";Lpeek(Inter+22);Tab(40);
  Print "Endadresse DMA              :";Lpeek(Inter+26)
  Print "Gelesene/geschriebene Bytes:";Dpeek(Inter+8)
Return
' Prozedur Readprog
' Liest Programm aus Datazeilen ein
'
Procedure Readprog(L)
  Prg$=Space$(L)
  T=1
  Repeat
    Read A$
    If A$<>"***"
      @Hextodec(Upper$(A$))
      Mid$(Prg$,T,1)=Chr$(S)
    Endif
    Inc T
  Until A$="***"
Return
'
' Prozedur Hextodec
' Wandelt A$ in Dezimal um (Ziel: S)
'
Procedure Hextodec(A$)
  S=0
  A$=Upper$(A$)
  For I=1 To Len(A$)
    A=Asc(Right$(A$,I))
    If A>64
      A=A-7
    Endif
    A=A-48
    S=S+A*16^(I-1)
  Next I
Return
'
'
' Sektormon: Kleiner Diskmon mit Standardfähigkeiten
'
Procedure Sektormon
  Cls
  Prn%=0
  Status%=0
  Seite%=0
  Track%=0
  Sector%=1
  Drive%=0
  Sec$=Space$(512)
  R$=Chr$(27)+"p"
  O$=Chr$(27)+"q"
  @Lesen
  Do
```

```
    Print
    Print "******* Minimon (C) 1987 Claus Brod *** Status: ";Status%;" ******"
    Print R$;"R";O$;"ead, ";R$;"W";O$;"rite, ";R$;"T";O$;"rk (";Track%;"), ";
    Print R$;"S";O$;"eite (";Seite%;"), Se";R$;"k";O$;"tor (";Sector%;"), ";
    Print R$;"E";O$;"dit, ";R$;"D";O$;"ump, ";R$;"P";O$;"rn (o";
    If Prn%
      Print "n), ";
    Else
      Print "ff), ";
    Endif
    Print "E";R$;"x";O$;"ec, ";R$;"Q";O$;"uit"
    M$=Upper$(Input$(1))
    If M$="X"
      @Exe
    Endif
    If M$="P"
      Prn%=1-Prn%
    Endif
    If M$="T"
      Input "Track";Track%
    Endif
    If M$="S"
      Input "Seite";Seite%
    Endif
    If M$="K"
      Input "Sektor";Sector%
    Endif
    If M$="R"
      @Lesen
    Endif
    If M$="W"
      @Schreiben
    Endif
    If M$="E"
      @Edit
    Endif
    If M$="D"
      @Dump(512)
    Endif
    Exit If M$="Q"
  Loop
Return
'
'
' Sektor einlesen
'
Procedure Lesen
  Buffer%=Varptr(Sec$)

  Status%=Xbios(8,L:Buffer%,L:0,Drive%,Sector%,Track%,Seite%,1)
Return
'
'
' Sektor schreiben
'
Procedure Schreiben
  Buffer%=Varptr(Sec$)
  Status%=Xbios(9,L:Buffer%,L:0,Drive%,Sector%,Track%,Seite%,1)
Return
'
'
' Sektorpuffer ausgeben
'
Procedure Dump(L)
  If Prn%=0
    Open "O",#1,"con:"
  Else
    Open "O",#1,"prn:"
  Endif
  Print #1,"Track: ";Track%;" Sektor: ";Sector%;" Seite: ";Seite%
  For T=1 To L Step 16
    A$=Str$(T-1)
    While Len(A$)<3
      A$="0"+A$
    Wend
    D$="   "
    Print #1,A$;D$;
    For I=0 To 15
      V$=Mid$(Sec$,T+I,1)
      A$=Hex$(Asc(V$))
      If V$<" " Or V$>"z"
        V$="."
      Endif
      Print #1,Right$("0"+A$,2)'
      D$=D$+V$
    Next I
    Print #1,D$
    A$=Inkey$
    If A$=" "
      A=Inp(2)
    Endif
    Exit If A$>"" And A$<>" "
  Next T
  Close #1
Return
'
'
```

Fortsetzung von Listing 1

Finger (sorry, auf die Bits) schauen können, zum anderen zeigt sie ausführlich an, was die Fehler/Statuscodes im DMA- und FDC-Status bedeuten.

Nun eine Liste der Funktion des Trackmon und Erläuterungen dazu:

„0"
Diese Funktion schickt einen Restore-Befehl an den Diskcontroller. Dieser fährt dann den Lesekopf des Laufwerks auf die Spur 0 zurück.

„1"
STEP läßt den Lesekopf eine Spur weit in die bisher eingeschlagene Richtung fahren.

„2"
STEP-IN fährt den Lesekopf eine Spur nach innen, zu höheren Tracknummern hin.

„3"
STEP-OUT fährt den Lesekopf eine Spur nach außen.

„5"
RD SEC liest einen Sektor auf der Spur, der im Moment eingestellt ist. Dazu müssen Sie dem Trackmon sagen, welchen Sektor und wieviel Bytes Sie lesen wollen. Das können Sie benutzen, um einen Sektor nur zum Teil oder auch Sektoren mit 1024 Bytes zu lesen (auch das gibt es!), bzw. auch

„4"
SEEK fährt eine bestimmte Spur an, die Sie eingeben müssen.

```
' Byte ändern
'
Procedure Edit
  Input "Byte Nr.";Byte%
  Print "Alter Wert ist: ";Asc(Mid$(Sec$,Byte%+1,1))
  Input "Neuer Wert in dezimal oder ASCII";Wert$
  Wert%=Val(Wert$)
  If Wert%=0 And Left$(Wert$,1)<>"0"
    Wert%=Asc(Wert$)
  Endif
  Mid$(Sec$,Byte%+1,1)=Chr$(Wert%)
Return
'
'
' anderes Programm ausführen
'
Procedure Exe
  Fileselect "*.*","",Sel$
  If Exist(Sel$)<>0
    S=Fre(0)
    Reserve 50000
    Showm
    Exec 0,Sel$,"",""
    Reserve S-1000
  Endif
Return
'
'
' Datawüste mit den Daten für LOCKSLEY und SELECT
' sowie ein paar Trackdaten
'
Locksley:
Data 60,24,0,0,0,0,0,0,0,0,0,0,0,0,0,0
Data 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
Data 0,0,0,0,0,0,48,E7,FF,FE,42,80,61,0,1,36
Data 45,FA,FF,EE,24,80,3C,3A,FF,CA,45,FA,FF,E4,34,BC
Data 0,0,45,FA,FF,DA,34,BC,0,0,50,F9,0,0,4,3E
Data 61,0,1,1E,51,F9,0,0,4,3E,45,FA,FF,C4,20,12
Data 61,0,1,2,4C,DF,7F,FF,4E,75,32,3C,0,1E,61,A
Data 33,C7,0,FF,86,4,32,3C,0,1E,51,C9,FF,FE,4E,75
Data 2E,3C,0,4,0,0,8,39,0,5,0,FF,FA,1,67,60
Data 53,87,67,40,45,FA,FF,8E,4A,52,67,EA,45,FA,FF,70
Data 15,79,0,FF,86,9,0,1,15,79,0,FF,86,B,0,2
Data 15,79,0,FF,86,D,0,3,20,3A,FF,54,B0,BA,FF,5C
Data 6D,C4,1E,3C,0,D0,61,0,0,EC,45,FA,FF,58,34,BC
Data 0,0,60,1C,61,1A,1E,3C,0,D0,61,0,0,D8,45,FA
Data FF,3E,34,BC,0,1,45,FA,FF,3C,34,BC,0,0,4E,75
Data 30,39,0,FF,86,4,45,FA,FF,1A,34,80,4E,75,30,39
Data 0,FF,86,6,45,FA,FF,E,34,80,42,41,12,39,0,FF
Data 86,9,E1,89,12,39,0,FF,86,B,E1,89,12,39,0,FF
Data 86,D,45,FA,FE,F6,24,81,92,BA,FE,EC,45,FA,FE,DA
Data 34,81,4E,75,45,FA,FE,E0,24,87,13,C7,0,FF,86,D
Data E0,8F,13,C7,0,FF,86,B,E0,8F,13,C7,0,FF,86,9
Data 2E,3A,FE,C4,42,80,30,3A,FE,B0,DE,80,45,FA,FE,BC
Data 24,87,4E,75,2F,0,3F,3C,0,20,4E,41,5C,8F,4E,75
Data 3E,6,3A,6,8,6,0,7,66,10,CC,3C,0,F0,BC,3C
Data 0,10,67,0,1,64,66,0,1,50,8,6,0,6,66,A
Data 8,6,0,5,67,2A,66,0,0,F4,CC,3C,0,F0,BC,3C
Data 0,C0,67,0,1,66,BC,3C,0,E0,67,6A,BC,3C,0,F0
Data 67,0,0,9E,61,0,FE,B4,32,3C,0,FA,60,0,FE,BC
Data 2E,3A,FE,48,61,0,FF,6E,45,FA,FE,5A,34,BC,0,1
Data 33,FC,0,90,0,FF,86,6,33,FC,1,90,0,FF,86,6
Data 33,FC,0,90,0,FF,86,6,3E,3C,0,E,61,0,FE,7C
Data 33,FC,0,84,0,FF,86,6,3E,3A,FE,C,61,0,FE,6C
Data 33,FC,0,80,0,FF,86,6,3E,5,61,0,FE,5E,61,0
Data FE,70,60,0,FE,EA,2E,3A,FD,F2,61,0,FF,18,45,FA
Data FE,4,34,BC,0,1,33,FC,0,90,0,FF,86,6,33,FC
Data 1,90,0,FF,86,6,33,FC,0,90,0,FF,86,6,3E,3C
Data 0,E,61,0,FE,26,33,FC,0,80,0,FF,86,6,60,B8
Data 2E,3A,FD,B8,61,0,FE,DE,45,FA,FD,CA,34,BC,0,1
Data 33,FC,1,90,0,FF,86,6,33,FC,0,90,0,FF,86,6
Data 33,FC,1,90,0,FF,86,6,3E,3C,0,E,61,0,FD,EC
Data 33,FC,1,80,0,FF,86,6,60,0,FF,7E,2E,3A,FD,7C
Data 61,0,FE,A2,45,FA,FD,8E,34,BC,0,1,33,FC,1,90
Data 0,FF,86,6,33,FC,0,90,0,FF,86,6,33,FC,1,90
Data 0,FF,86,6,3E,3C,0,E,61,0,FD,B0,33,FC,1,84
Data 0,FF,86,6,3E,3A,FD,40,61,0,FD,A0,33,FC,1,80
Data 0,FF,86,6,60,0,FF,32,33,FC,0,80,0,FF,86,6
Data 61,0,FD,88,60,0,FD,9A,33,FC,0,86,0,FF,86,6
Data 3E,3A,FD,12,61,0,FD,74,33,FC,0,80,0,FF,86,6
Data 3E,5,61,0,FD,66,60,0,FD,78,2E,3A,FC,FE,26,7A
Data FC,FE,61,0,FE,20,33,FC,0,90,0,FF,86,6,33,FC
Data 1,90,0,FF,86,6,33,FC,0,90,0,FF,86,6,3E,3C
Data 0,1,61,0,FD,36,33,FC,0,80,0,FF,86,6,34,3A
Data FC,C8,3E,5,61,0,FD,24,61,0,FD,36,32,3A,FC,C4
Data 36,C1,51,CA,FF,EE,60,0,FD,A6,0,0
Data ***
'
Select:
Data 60,E,0,0,0,0,0,0,0,0,0,0,0,0,0,0
Data 48,E7,FF,FE,42,80,61,6A,45,FA,FF,F2,24,80,50,F9
Data 0,0,4,3E,3E,3A,FF,DC,66,14,33,FC,0,80,0,FF
Data 86,6,32,39,0,FF,86,4,8,1,0,7,66,F4,C,7
Data 0,8,6C,5C,A,7,0,7,CE,3C,0,7,40,E7,0,7C
Data 7,0,13,FC,0,E,0,FF,88,0,10,39,0,FF,88,0
Data C0,3C,0,F8,8E,0,13,C7,0,FF,88,2,51,F9,0,0
Data 4,3E,46,DF,45,FA,FF,96,20,12,61,6,4C,DF,7F,FF
Data 4E,75,2F,0,3F,3C,0,20,4E,41,5C,8F,4E,75,51,C9
Data FF,FE,4E,75,2F,A,3F,3C,0,9,4E,41,5C,8F,4E,75
Data 45,FA,0,6,61,EE,60,CC,52,6F,75,74,69,6E,65,20
Data 6E,6F,63,68,20,6E,69,63,68,74,20,69,6D,70,6C,65
Data 6D,65,6E,74,69,65,72,74,2E,D,A,0,0
Data ***
'
Trackdaten:
Data 4E,2
' Trackvorspann
'
Data 00,2
' Sektorvorspann
Data F5,3,FE,1
' Syncbytes und Adreßmarke
Data Track,1,Side,1,1,1,3,1,F7,1
' Sektorvorspann und Checksumme
Data 4E,14,0,A
' Lückenbytes
Data F5,3,FB,1
' Syncbytes und Datenmarke
Data CB,400,F7,1
' Sektordaten und Checksumme
Data 4E,10
' Lückenbytes
'
Data 00,2
' Sektorvorspann
Data F5,3,FE,1
' Syncbytes und Adreßmarke
Data Track,1,Side,1,2,1,0,1,F7,1
' Sektorheader (Sektor mit 128 Bytes) und Checksumme
Data 4E,14,0,A
' Lückenbytes
Data F5,3,FB,1
' Syncs und Datenmarke
Data CB,40,F7,1
' Sektordaten (unvollständig) und Checksumme
Data 4E,10
' Lückenbytes
'
Data 00,2
' Sektorvorspann
Data F5,3,FE,1
Data Track,1,Side,1,3,1,2,1,F7,1
' normaler 512-Byte-Sektor
Data 4E,14,0,A
Data F5,3,FB,1
Data CB,200
' Sektordaten
Data F7,1,4E,10
'
Data 00,2
' Sektorvorspann
Data F5,3,FE,1
Data Track,1,Side,1,4,1,4,1,F7,1
' Sektor mit der Größenangabe '4'
Data 4E,14,0,A
Data F5,3,FB,1
Data 02,200
' Sektordaten des zweiten Sektors Nr. 3
Data F7,1,4E,10
'
Data 4E,200
' Tracknachspann
Data ***,***
```

Fortsetzung von Listing 1

mehrere Sektoren hintereinander zu lesen. Der Inhalt der eingelesenen Sektorn kommt in den String buf$ und kann mit der Funktion „D" angezeigt werden.

„6"
WR SEC arbeitet ähnlich wie RD SEC. Mit dieser Funktion können Sie auch Sektoren schreiben, die nicht betriebssystemkonform sind; solche mit 1024 Bytes, halbe oder gleich mehrere Sektoren usw. Sie müssen die Sektornummer angeben, mit der Sie beginnen wollen,sowie die Anzahl der Bytes, die geschrieben werden sollen. Tatsächlich geschrieben wird der Inhalt von buf$, also das, was Sie vorher mit einer Lesefunktion eingelesen und evt. per Edit verändert haben. Aus irgendwelchen Gründen müssen Sie beim Schreiben immer $20 (=32) Bytes mehr angeben, als in Wirklichkeit geschrieben werden sollen, eine Eigenheit des DMA-Controllers, die ich mir nicht recht erklären kann (sie gilt auch für WR TRK). Aber wenn Sie dies berücksichtigen, funktioniert die Sache einwandfrei.

„7"
RD ADR liest den Vorspann eines Sektors ein. Er besteht aus Tracknummer, Seitennummer (0=Vorderseite, 1=Rückseite), Sektornummer, Größe des Sektors (0=128 Bytes, 1=256 Bytes, 2=512 Bytes, 3=1024 Bytes) und zwei Bytes Checksumme. Sie müssen angeben, wieviel Adreßfelder (so nennt man diesen Vorspann auch) Sie holen wollen. Beachten Sie hier, daß der DMA-Controller bis zu 16 Bytes intern lagert und diese gerne vergißt. Im zweiten Pufferstring, buf2$, archiviert der MINIMON (bzw. LOCKSLEY) den jeweiligen Controllerstatus pro Adreßfeld, das sind pro Adreßfeld zwei Bytes, von denen das obere unwichtig ist.

„8"
RD TRK liest einen kompletten Track in buf$ ein. Vorher wird noch gefragt, wieviel Bytes dieser Spur transferiert werden sollen. Sie werden des öfteren bemerken, daß recht merkwürdige Ergebnisse beim Lesen entstehen. Das liegt daran, daß der Read-Track-Befehl

im Controller selbst nicht einwandfrei funktioniert!

„9"
WR TRK schreibt eine ganze Spur, formatiert sie also. Diese Funktion habe ich bewußt recht flexibel gehalten, damit Sie beliebige Diskettenformate erzeugen können. Beim Formatieren wird der Inhalt von T$ in der Länge, die Sie angeben, auf die Spur geschrieben. Beim Write-Track-Kommando ist aber noch anzumerken, daß alle Bytes über $F5 spezielle Kommandos sind. Schauen wir uns genau an, welche Bytes man schreiben muß, um einen Sektor zu schreiben:

Sektorvorspann (einige Nullbytes)

Drei Syncbytes
(erzeugt man durch drei $F5-Befehlsbytes)

Adreßmarke
(kündigt ein Adreßfeld, also den Vorspann des Sektors an und besteht aus einem einzigen $FE-Byte)

Adreßfeld
(besteht aus Angabe der Tracknummer, Seitennummer, Sektornummer und Sektorgröße, siehe RD ADR)

Checksumme
(erzeugt man durch das Befehlsbyte $F7)

Lückenbytes
(etwa 30 davon, meist etwa 20 $4E-Bytes und danach etwa 10 Nullbytes)

Drei Syncbytes
(zu erzeugen mit $F5-Befehlsbytes)

Datenmarke
(einmal $FB schreiben)

Sektordaten
(128, 256, 512, 1024 Bytes, je nach Adreßfeld)

Checksumme
(einmal $F7 schreiben)

Sektornachspann
(etwa 10 Lückenbytes, etwa $4E's)

Bleibt anzumerken, daß in den Sektordaten selbst kein Byte über $F5 auftreten darf. Um die Trackformatierung einfach zu machen, liest die Prozedur WR TRK die Daten von dem Label 'Trackdaten:' ein und interpretiert sie als Anweisungen, wie sie den Track erstellen soll. Das Format dieser Data-Zeilen ist „zu schreibendes Byte" (in Hexadezimalcode), „ Anzahl" (das heißt: wie oft soll das angegebene Byte geschrieben werden; auch diese Zahl muß in Hexadezimalcode angegeben werden). Es gibt spezielle „Bytes" in diesen Zeilen, die besondere Bedeutung haben. Schreibt man statt eines Bytes einfach „Track", wird der aktuelle Track eingesetzt. „Side" fügt die aktuelle Seitennummer ein. „★★★, ★★★" kennzeichnet das Ende der Daten. Ein Beispiel-Track ist schon angegeben, der eine wilde Kombination von Sektoren auf die Spur schreibt, unter anderem auch einen Sektor, der nicht vollständig ist, aber durchaus Daten enthalten kann.

„A"
IRQ unterbricht den Floppycontroller bei seiner momentanen Arbeit. Da aber der Controller eh schon fix und fertig ist, wenn man im Menü gelandet ist, darf man diesen Menüpunkt nur als „der Vollständigkeit und Demo halber" verstehen. Danke.

„D"
DUMP BUF listet einen Puffer aus. Vorher wird man gefragt, welcher Puffer angezeigt werden soll. Puffer 2 ist nur nach RD ADR gefüllt, alle anderen Lesebefehle kleistern Puffer 1 zu. SPACE, während des Listens hält die Ausgabe an, jede andere Taste bricht das Listen ab. Nach dem Ende der Ausgabe wird man gefragt, ob man den Puffer edieren will. Wenn ja, wird man (wie bei der Edit-Option des Sektormon) gefragt, welches Byte man ändern möchte und welchen Wert man dafür einsetzen will.

„R"
REGISTER, wird in der nächsten ST ergänzt.

„S"
SEITE, wechselt die aktuell eingestellte Seite. War sie vorher 0 (Vorderseite), so ist sie nachher 1 (Rückseite). Und umgekehrt.

„P"
schaltet den Drucker an und aus (wie beim Sektormon). DUMP BUF leitet die Ausgabe entsprechend um.

„Q"
QUIT, dürfte klar sein – Ausstieg in die Auswahl Sektormon/Trackmon/Quit.

Schließlich noch ein paar Tips, was Sie ausprobieren könnten. Schreiben Sie mal den im Listing angegebenen Beispieltrack auf eine Diskette (die aber möglichst leer sein sollte, sonst verfluchen Sie mich noch). Versuchen Sie danach, im Trackmonitor die Sektoren einzulesen, auch den falsch geschriebenen zweiten (er ist mit 128 Bytes angekündigt, es sind aber nur 64 Bytes geschrieben). Das wird ihnen ohne Probleme gelingen, wenn Sie die richtige Bytezahl angeben. Versuchen Sie es danach mit dem Sektormon. Na? Sie bekommen einen Status von –11. Das Betriebssystem wäre an dieser Stelle schon ausgestiegen.

Ach ja, diese Fehlercodes im Sektormon sollte ich vielleicht auch noch nachtragen:

Status	Fehler
0	alles klar
–1	nicht genau bestimmter Fehler
–2	Laufwerk nicht bereit
–3	Befehl nicht bekannt
–4	Prüfsummenfehler
–5	Befehl so nicht gültig
–6	Kann Track nicht finden
–7	Bootsektor ziemlich kaputt
–8	Kann Sektor nicht finden
–9	nicht genau bestimmter Fehler
–10	Fehler beim Schreiben
–11	Fehler beim Lesen
–12	nicht genau bestimmter Fehler
–13	Schreibschutz an
–14	Diskette gewechselt
–15	Kann Gerät nicht ansprechen
–16	Verify Error
–17	Keine Diskette eingelegt

Diese Fehlercodes haben aber nichts (oder nur wenig) mit dem Statusbyte des Floppycontrollers zu tun.

Das Betriebssystem (und ältere Kopierprogramme) kopieren halbe Sektoren nicht oder nur fehlerhaft. In einem Kopierschutz könnten Sie nun abfragen, ob der halbe Sektor so vorhanden ist, wie Sie ihn geschrieben haben. Wenn nicht – BRTZL!

Sie können auch mal probieren, was passiert, wenn Sie den Abstand zwischen Sektorvorspann und Sektor selbst sehr groß machen. Der Sektormon meldet auch hier einen Fehler, obwohl eigentlich der ganze Sektor da ist!

Oder schreiben Sie mal zwei Sektoren mit den gleichen Sektornummern. Ein „dummes" Kopierprogramm kopiert garantiert nur einen der beiden. Sie aber prüfen in Ihrem Programm, ob das doppelte Lottchen da ist, mit den beschriebenen Konsequenzen.

Im Grunde können Sie mit Hilfe der Befehle READ ADDRESS, READ TRACK und READ SECTOR, die

Sie jetzt im EXTENDED MINIMON zur Verfügung haben, jeden Kopierschutz zumindest analysieren; wenn Sie sich ein bißchen anstrengen, können Sie ihn dann auch kopieren. Das wird natürlich manchen Softwarehersteller gar nicht gefallen, daß jetzt – dank MINIMON – jeder ST-User in seiner Software Einblick hat. Deswegen mein Appell: Kopierschutz knakken als „Sport" ist OK, aber Raubkopieren... Doch was ein hartgesottener Kopierer ist, wird ob dieser Zeilen auch nur in Gelächter ausbrechen. Leider. Andererseits erhoffe ich mir von der allgemeinen Verfügbarkeit von Werkzeugen wie MINIMON, daß die Hersteller den Kopierschutz irgendwann einmal ad acta legen, zur Freude der rechtmäßigen Anwender. Seriennummern in den Programmen sind sowieso viel sicherer und ungleich schwieriger zu finden als ein Diskettenkopierschutz.

Lassen Sie sich davon nicht betrüben, probieren Sie selbst neue Ideen aus (wie wär's mit Manipulationen des Sektorvorspanns...). In der nächsten ST stelle ich die Controllerbefehle detailliert vor, dann kommt sicher noch ein größerer Ideenschwall auf Sie zu.

WANDLUNGSFÄHIG

PRGTOLST.BAS ist ein klitzekleines Utility in GfA-Basic, das mit dem AS68 assemblierte (relokatible!) Programme in Datazeilen wandelt, die dann per 'MERGE' in Ihr BASIC-Programm eingefügt werden können.

Beim Start fragt PRGTOLST in einer Fileselect-Box nach dem Namen der zu wandelnden Datei. Später müssen Sie noch die Größe der Datei eingeben, welche Sie aus dem Verzeichnis des Desktop erfahren (Textanzeige einschalten). PRGTOLST gibt zuerst alle Bytes der Datei,dann die Datazeilen auf dem Bildschirm an.Die Datazeilen werden gleichzeitig in eine Datei geschrieben, die den selben Namen wie die Originaldatei hat, zusätzlich aber die Extension LST besitzt. Diese Datei können Sie dann MERGEn.

Für den Experten: PRGTOLST.BAS schneidet den Programmheader eines fertig gelinkten Programms (das sind die ersten 28 Bytes) ab, weil er für ein vollständig relokatibles Programm, das in BASIC eingebunden wird, nicht notwendig ist.

```
Claus Brod

' PRGTOLST
' Konvertiert PRG-File zum LST-File, das
' mit MERGE eingeladen werden kann
' Written 1987 by Claus Brod, Am Felsenkeller 2,
'                  8772 Marktheidenfeld, 09391/3206
'
'
Dim A%(5000)                                    ! maximale Filegröße
Fileselect "\*.*","",F$
Print F$
Input "Größe der Datei in Bytes";G              ! Größe der Datei
S=Varptr(A%)
'
Bload F$,S                                      ! Datei in Integerarray lesen
'
' Datei kurz ausgeben
'
Print "printing hex values of ";F$
For T=0 To G
  P=Peek(T+S)
  P$=Hex$(P)
  Print Left$("00",2-Len(P$));P$'
  If Inkey$=" "
    Void Inp(2)
  Endif
Next T
'
' Datei auf Diskette (aktuelles Verzeichnis)
' schreiben
'
Print
A=Instr(F$,".")
G$=Left$(F$,A)+"lst"
Print "Creating data lines for ";G$
Open "O",#1,G$
For T=28 To G Step 16
  Print #1,"data ";
  Print "data ";
  For I=0 To 15
    P$=Hex$(Peek(T+S+I))
    Print #1,P$;
    Print P$;
    If Inkey$=" "
      Void Inp(2)
    Endif
    If I<15
      Print #1,",";
      Print ",";
    Endif
  Next I
  Print #1
  Print
Next T
Close #1
```

Listing 2: PRGTOLS1.Bsc.

Bei der Einbindung eines solchen Programmes geht man vor, indem man in BASIC einen genügend langen String definiert (etwa: PRG$=SPACE$(1000)), sich davon die Anfangsadresse geben läßt (Anfang=Varptr(prg$)) und daraufhin das Maschinenprogramm in einer Schleife in diesen String schreibt. (siehe auch die Routine Readprog im „TRACKMON"). Im „TRACKMON" habe ich mich dafür entschieden, das Maschinenprogramm aus Data-Zeilen zu entnehmen, damit „ Nur-BASIC-Programmierer" die Chance haben, diesen Teil des MINIMON zu nutzen. Die so erzeugte Kopie des Maschinenprogramms im Speicher kann man schließlich mit Call (Anfang) aufrufen. In meinen Maschinenprogrammen bin ich aus Effizienzgründen davon abgekommen, die nötigen Parameter mit dem Call-Befehl übergeben zu lassen. Die Parameter werden in festgelegten Bereichen des Maschinencodes geschrieben (siehe Handhabung von LOCKSLEY.S und SELECT.S im „TRACKMON", vor allem in der Routine Prntable) und auch von dort wieder abgeholt. (C.B.)

Der Fehlerteufel geht um!!!

In der September-Ausgabe haben sich beim Montieren der Zeitung leider einige Fehler eingeschlichen. Wir bitten dies zu entschuldigen.

Im Listing von LOCKSLEY.S sind Zeilen verschluckt worden. Auf Seite 3 fehlt am Ende

```
move.w d1,(a2)
rts
```

Auf Seite 5 fehlt unten

```
move.w #$190,dmodus
move.w #14,d7
```

ATARI ST ASSEMBLER-BUCH
Markt & Technik Verlag
Haar bei München 87
299 Seiten
Preis: DM 59,—

Obwohl für den ATARI ST mittlerweile fast alle gängigen Hochsprachen verfügbar sind, übt die Assemblerprogrammierung immer noch einen magischen Reiz aus. So mancher BASIC oder PASCAL Programmierer spielt mit dem Gedanken, es einmal mit der Maschinensprache des 68000er zu probieren oder zumindestens sein BASIC bzw. PASCAL Programme durch kleine Maschinenspracheroutinen zu beschleunigen. An Literatur zum Thema Maschinensprache des 68000er mangelt es sicher nicht, doch nur sehr wenige Bücher beschäftigen sich auch gleichzeitig mit der Hardware des ATARI ST. Eines dieser Bücher ist das ATARI ST ASSEMBLER-BUCH von P. Wollschläger, erschienen bei Markt und Technik.

„Ganz schnell zur Praxis" ist die Überschrift des vierten Kapitels. Dieses Motto zieht sich als roter Faden durch das gesamte Buch. So hält sich der Autor nicht mit langen Einführungen in die Grundlagen auf. Das Minimum an Theorie, das für einen ersten Einstieg in die „Unterwelt" des Rechners ausreichen sollte, wird auf den ersten 50 Seiten im Schnelldurchgang abgehandelt. Dabei geht es um u. a. um folgende Themen: Zahlensysteme, Sinn und Zweck eines Assemblers, die Architektur des 68000er. Positiv ist anzumerken, daß am Ende des ersten Kapitels alle gängigen Assembler für den ST vom Autor kurz vorgestellt und bewertet werden.

Danach kann der Leser sein erstes Assemblerprogramm eintippen. Er erhält, wenn er das abgebildete – jedoch bewußt unvollständige – Programm richtig eingegeben hat, das Wort 'HALLO' auf dem Bildschirm. Während andere Bücher an dieser Stelle meisten irgendwelche trockenen Multiplizier- oder Sortierroutinen enthalten, stellt der Autor das Innen leben des ST anschaulich vor. Der Leser lernt an zahlreichen Betriebssystemaufrufen die Anwendung der wichtigsten Befehle des 68000er kennen. PROGRAMMIEREN IN ASSEMBLER ist kein akademisches Lehrbuch, das systematisch die Maschinensprache des 68000er abhandelt. Es ist vielmehr ein Kochbuch, das geradezu dazu einlädt, den Assembler zu laden und Programmierrezepte auszuprobieren. Die anschaulich erklärten Beispiele, zu denen u. a. eine vollautomatische RAM Disk und ein Diskettenmonitor gehören, demonstrieren sehr gut die Wirkung der Maschinenbefehle und machen Appetit auf mehr. Nach dem Durcharbeiten der Kapitel 7 und 8 sind dem Leser Begriffe wie Include Files, DTA-Buffer oder FAT genauso geläufig wie einem BASIC Programmierer die Begriffe Schleifen oder Unterprogramme. Die in Kapitel 9 vorgestellten Grafikroutinen zeigen sehr gut, wie Maschinensprache auf dem 68000er sinnvoll eingesetzt werden kann. Allerdings hätte dieses Kapitel durchaus umfangreicher ausfallen können. Auch Kapitel 15 mit dem Titel „Die Tricks der Profis" ist mit nur neun Buchseiten wenig attraktiv.

Am Anfang des Buches rät der Autor dem Leser, wohl in erster Linie jenen, die bereits erste Erfahrungen auf CPU's wie der 6502 oder dem Z80 gesammelt haben, alles zu vergessen, was sie bisher über Maschinenspracheprogrammierung wissen. Alles sollte man allerdings nicht vergessen, denn das Tempo, mit dem der Autor sein Wissen in einem durchweg lockeren und saloppen Stil an den Leser weitergibt, ist an manchen Stellen ein wenig zu rasant. Mehr Grundlagenwissen über binäre Verknüpfungen oder die grundsätzlichen Eigenschaften der Maschinensprachprogrammierung ist an manchen Stellen des Buches von Nöten.

Fazit:

Das ATARI ST ASSEMBLER-BUCH kann jedem ATARI Freak mit gutem Gewissen als Pflichtlektüre empfohlen werden. Das Schöne ist, daß man mit dem Kauf des Buches nicht nur eine sehr gut gelungene Einführung in die Maschinensprache des 68000er erhält, sondern zusätzlich eine Fülle an nützlichen Informationen über das Innenleben seines ATARI ST bekommt.

(PM)

GFA Handbuch TOS & GEM

GFA Systemtechnik
Düsseldorf 87
370 + 30 Seiten
Preis DM 49,-

Dieses Buch hat, obwohl es von GFA verlegt wird, keine Gemeinsamkeiten mit GFA Basic. Es bezieht sich auf annäherend alle rechnerinternen Dinge, welche den ST betreffen.

Auf ca. 50 Seiten werden alle Bausteine und Schnittstellen der Hardware beschrieben. Der Hauptteil des Buches beschäftigt sich mit der Programmierung des Betriebssystems, sowie dem GEM. Jede Funktion und deren Parameter und deren Wirkungsweisen sind aufgeführt. Weiterhin enthält das Buch ST Systemadressen; um einige zu nennen, die Line-A-Variablen und Tastaturprozessorkommandos. Beispielprogramme sind nicht aufgeführt, jedoch ein Indexheft, welches wichtige Informationen in Kurzform enthält, ist beigefügt.

Insgesamt wird das Buch als Nachschlagewerk empfohlen. Man erhält auf 350 Seiten Informationen, für welche man sich jedoch schon mit der Materie befaßt haben sollte.

Ein empfehlenswertes Nachschlagewerk für Fortgeschrittene.

HS

Ein Case für alle Fälle

Case Of Struktur in OMIKRON. Basic

Eigene Prozeduren oder Funktionen sind in Basic kein Problem; aber, daß man auch völlig neue Strukturen einbauen kann, ist nicht vorgesehen. Das folgende Beispiel zeigt, daß es trotzdem geht.

Pascal bietet die Struktur CASE OF. Das ärgerte natürlich die fortschrittlichen Basic-Programmierer schon lange. Das Resultat: es geht doch. Das Prinzip ist relativ einfach. Zuerst erfolgt ein Befehl Case (Variable). Dahinter verbirgt sich nichts anderes als eine Prozedur namens Case. Der Ausdruck in Klammern ist der Übergabeparameter. Dieser wird innerhalb des Unterprogrammes einer bestimmten Variablen zugeordnet, die später noch Verwendung findet. Dann folgt die Anweisung 'Of(Wert)', welche prüft, ob die Variable hinter 'Case', mit diesem Wert übereinstimmt. Ist dies der Fall, wird die folgende Programmzeile ausgeführt. Ist dies nicht der Fall, so wird sie übersprungen und in der übernächste Zeile fortgefahren. Auch 'Of' ist eine Prozedur. Ihre Aufgabe besteht in nichts anderem, als den Parameter hinter OF mit dem hinter Case zu vergleichen. Dementsprechend wird ein Flag gesetzt oder gelöscht.

Der Witz der Sache kommt mit der Anweisung 'Do'. 'Do' ist nämlich keine Prozedur, sie darf es auch nicht geben. So löst der Interpreter beim Bearbeiten dieser Anweisung eine Fehlermeldung aus, und zwar Fehler Nummer 8. Das ist nun die Lösung. In der Fehlerroutine, initialisiert mit ON ERROR GOTO, wird die Fehlernummer identifiziert und, je nachdem ob das gerade erwähnte Flag gesetzt oder gelöscht ist, das Programm bei der nächsten oder übernächsten Zeile fortgesetzt.

Dadurch entsteht leider zwangsweise eine Einschränkung: Nach einem DO darf nur eine Zeile folgen; plagt einen aber trotzdem das Verlangen nach mehreren, so gibt man einfach in der folgenden Zeile ein weitere DO ein und darunter die zweite Befehlszeile, usw.

Die neue Struktur wurde also durch Einschleusung eines absichtlichen Fehlers geschaffen. Dies mag zwar seltsam klingen, doch geschieht das gerade beim ST sehr häufig. Man denke nur an die LINE A Traps, wobei der Prozessor auch durch ungültige Befehle in eine Fehlerroutine gelangt, wo er den Fehler analysiert und dementsprechend ein Unterprogramm ausführt. Die Idee ist also garnicht so neu, aber wirkungsvoll ist sie dennoch.

Die Syntax

```
Case (Variable)
  Of (Wert1):Do
     Anweisungszeile1
  Of (Wert2):Do
     Anweisungszeile2
     Do
     Anweisungszeile2a
  Of (Wert3):Do
     Anweisungszeile3
'End Case
```

```
'CASE-Struktur - selbstdefiniert in OMIKRON.BASIC
'Zur Verwendung von Case-Strukturen bitte in's eigene Programm einbauen
'  - Prozedur Case
'  - Prozedur Of
'  - KEINESFALLS eine Prozedur mit dem Namen "Do" definieren!!!
'  - ON ERROR GOTO Fehler
'  - die 8 Zeilen von Label -Fehler bis zum zweiten ENDIF
'  - bitte darauf achten, daß die Zeilennummern aufeinanderfolgen (also
'    nicht 10,20,30..., sondern 1,2,3...) - am besten Zeilennummern
'    im Editor abschalten, OMIKRON.BASIC macht dann automatisch
'    Einser-Abstände.
'
'Funktionsweise
'  In der Prozedur "Case" wird lediglich der Stringausdruck zwischen-
'  gespeichert. Bei "Of" wird er mit dem bei "Of" angegebenen String-
'  ausdruck verglichen. Sind beide gleich, so wird das Flag "Case_Flag"
'  auf TRUE gesetzt, sonst auf FALSE.
'  Jetzt kommt der Trick: Die Prozedur "Do" existiert nicht! Also
'  entsteht, wenn man sie aufruft, ein Fehler (Undefined Statement,
'  Fehler Nr. 8). In der Fehler-Routine, die dank ON ERROR GOTO nun
'  durchlaufen wird, wird das Case Flag abgetestet: bei TRUE geht's
'  in der nächsten Zeile weiter (RESUME NEXT), sonst in der Über-
'  nächsten (RESUME ERL+2).

'Beispielprogramm

True=-1:False=0
ON ERROR GOTO Fehler
INPUT "Schwierigkeitsgrad (leicht,mittel,schwer) : ";Schwierigkeitsgrad$
Case( UPPER$( LEFT$(Schwierigkeitsgrad$,1)))
  Of("L"):Do
    PRINT "Nun, einfach fängt man an!"
  Of("M"):Do
    PRINT "Wer's weiters wagt, gewinnt auch mehr!"
  Of("S"):Do
    PRINT "Den feuerspein'den Drachen fürcht' der Recke nicht!"

'Ende der Case-Struktur. Der Rest bis END ist nicht notwendig.

PRINT : PRINT "Suche ihr drückendes Gefängnis und befreie die"
PRINT " Prinzessin aus der unverdienten Gefangenschaft!"
WAIT 3
REPEAT
  PRINT
  PRINT "Du bist in einer feuchten, dunklen Höhle."
  IF RND(3)=1 THEN PRINT "Wasser tropft von der Decke und sammelt sich in kleinen
                          Lachen auf dem Boden."
  PRINT "Richtungen: N,S,O,W"
  INPUT "Was willst du tun: ";Kommando$
UNTIL Player_Is_Mad
END

DEF PROC Case(V$)
Case_Vergleich$=V$
RETURN

DEF PROC Of(V$)
IF Case_Vergleich$=V$ THEN Case_Flag=True ELSE Case_Flag=False
RETURN

-Fehler
IF ERR =8 THEN ' Fehler 8="Undefined Statement". Prozedur "Do" ist undefiniert
   IF Case_Flag
      THEN RESUME NEXT
      ELSE RESUME ERL +2
   ENDIF
ELSE STOP
ENDIF
```

Case Of Struktur mit Beispiel

Des Kaisers neue Kleider oder Lattice C im neuen Gewand

Seit kurzer Zeit ist der Lattice C Compiler von Metacomco in einer neuen verbesserten Version erhältlich. Es handelt sich dabei um die Version 3.04, die mir zum Test vorlag.

Zum Lieferumfang des Compilers gehören drei Disketten sowie ein über 600 Seiten starkes Handbuch. Das Paket stellt ein vollständiges C-Entwicklungssystem dar und, enthält alle nötigen Hilfsprogramme, wie:

- 2 Pass C Compiler
- Linker
- sehr unmfangreiche Library
- Menu + Graphische Benutzer Shell zur einfachen Bedienung des Compilers
- GEM – Editor
- Symbolic Debugger mit Link-Loader
- Resource Construction Set
- Make Utility

Die neue Version des Compilers wurde um viele Details erweitert. So können nun Strukturen an Funktionen übergeben bzw. zurückerhalten werden und nicht nur Pointer auf diese. Außerdem wurden einige Beschränkungen der alten Version geändert. Der Compiler arbeitet in der neuen Version wesentlich flotter. Er kommt zwar nicht ganz an die Zeiten vom Megamax heran, aber man kann gut damit auskommen. Der Aufruf des Compilers gestaltet sich mittels eines Treiberprogramms (LC.TTP) relativ einfach. Es lassen sich jetzt auch Joker im Aufruf verwenden.

So compiliert das Kommando „LC F:s?" alle Sourcedateien auf Drive F: deren Name mit 's' beginnt und 2 Zeichen lang ist. Die Angabe der Extension ist nicht nötig. Der Compiler-Treiber hängt für Sourcedateien immer den Prefix '.C' an. Der Compiler läßt sich auch über den alten Batch Prozessor vom Entwicklungssystem steuern z. B. mit folgendem Batch File:

```
lc1.ttp -n %1
lc2.ttp %1
link.ttp %1 c -nolist
wait
```

Alle Änderungen gegenüber früheren Versionen sind im Handbuch erwähnt, wo eine Tabelle zu finden ist, die Auskunft darüber gibt, wann man einzelne Object Module neu compilieren muß, damit sie mit den neuen Library Funktionen einwandfrei arbeiten. Die Libraries wurden auch erheblich erweitert. Der Umfang lässt keine Wünsche offen. So sind die schnellen Floating Point Routinen von Motorola (Motorola 68881 wird optional unterstützt) ebenso enthalten, wie viele vom Unix Betriebssystem bekannte Funktionen, für String- und Filehand-

ling sowie alle Funktionen vom GEM-VDI bzw. GEM-AES. Zu den beiden letztgenannten ist allerdings eine kleine Unannehmlichkeit zu bemerken. Wie sicher viele von Ihnen wissen, arbeiten die GEM Funktionen auf dem ST mit Parametern einer Länge von 16 Bit. Bei den anderen gängigen Compilern für den ST wird für diese normalerweise der Datentyp 'int' (Integer) benutzt. Beim Lattice C ist eine Variable vom Typ 'int' nun aber 32 Bit lang, was die GEM Funktionen in arge Bedrängnis bringt und diese wiederum nicht selten mit einer Bombendrohungen auf diesen Mißstand aufmerksam machen. Wenn man aber seine Programme in Hinsicht auf Portabilität auslegt und in Anbetracht der Tatsache, daß unterschiedliche Computersysteme unterschiedliche Längen dieses Typs benutzen, ist diese Unzulänglichkeit leicht aus der Welt zu schaffen. Metacomco hat zu diesem Zweck eine Headerdatei mit einigen Definitionen zusammengestellt, die einem sehr hilfreich sind. Schreibt man statt 'int' 'WORD', bindet diese Headerdatei namens 'PORTAB.H' mit ein, so nimmt alles seinem gewohnten Lauf. Gelingt es nicht, muß es nicht am Compiler. Diese Tatsache hat auch ihre guten Seiten. So waren sauber geschriebene Programme nach erfolgtem Compilerlauf und unter Verwendung des LATTICE C auf dem IBM PC sofort und ohne Änderungen lauffähig. Es ist ein guter Compiler, mit dem mir das Arbeiten sehr viel Spaß gemacht hat.

Der Compiler erzeugt ein Objectcode File, wahlweise im GST oder DRI Format, das mit dem Linker bearbeitet werden kann, ohne Umweg über den Assembler, wie z.B. beim Compiler von Digital Research. Dies wäre nicht möglich, denn ein Assembler wird nicht mitgeliefert. Um eigene Assembler Routinen zu erstellen und in C-Programme mit einzubinden, kommt man um die Anschaffung eines Assemblers (der Assembler der Firma GST sei hierzu empfohlen) nicht herum.

über den Linker sollen hier keinen großen Worte verloren werden. Es handelt sich um das Produkt von GST, das auch bei diversen anderen Sprachen sein Können schon unter Beweis gestellt hat, z.B. Prospero Pascal, Prospero Fortran sowie natürlich auch bei dem Pascal Compiler von Metacomco. Leider arbeitet der Linker nicht mit der Geschwindigkeit, die der neue C Compiler vorlegt.

Im Folgenden werden nun noch kurz die wichtigsten Tools (Hilfsprogramme) vorgestellt, die dem Paket beiliegen.

Das Menu + Programm

Es handelt sich hierbei um eine Shell, mit der der Compiler sehr einfach gesteuert werden kann. Die Inbetriebnahme des Systems mit Hilfe dieses Programmes gestaltet sich sehr einfach. Es lassen sich alle wichtigen Parameter einstellen und abspeichern, die bei jedem Neustart automatisch geladen werden, so daß man stets seine eigene Shell zur Verfügung hat. Die Voreinstellungen auf der mitgelieferten Disk sind so ausgelegt, daß man ohne Änderungen sofort loslegen kann. Für Neulinge in Sachen 'Lattice C' ist im Handbuch ein kompletter Ablauf eines Edit-Compile-Linkprozesses (eines sogenannten „Turn Around") abgedruckt, den man Schritt für Schritt nachvollziehen kann, um zu sehen wie der Compiler und Linker arbeitet. Auch fertige Programme lassen sich von der Shell aus starten, ohne diese verlassen zu müssen. Bei aller Einfachheit, die diese Shell bietet, dürfen zwei Dinge nicht vergessen werden, die mir bei der Benutzung unangenehm auffielen. Zum einem lassen sich die einzelnen Funktionen der Shell 'nur' mit der Maus erreichen, was den Arbeitsfluß doch etwas bremst.

Den zweiten Punkt entdeckte ich, nachdem ich verzweifelt versucht hatte, die Shell über Tastaturkommandos zum Arbeiten zu bringen. Ich weiß ' Man lese zuerst das Handbuch dann...', naja ich bin halt ein ungeduliger Mensch. Nachdem ich also in der Shell einige Tastenkombinationen getestet hatte und nun wieder den Editor startete, wurden doch tatsächlich die in der Shell gedrückten Tasten in den Sourcecode übernommen. Da der Editor, wie wir später noch feststellen werden, im Gegensatz zu der Shell sehr gut über Tastatur gesteuert werden kann, können nun die seltsamsten Dinge passieren. Man stelle sich vor: es werden statt Buchstaben Befehle für 'Zeilen löschen' übergeben. Die Folge war ein verstümmelter Source Code. Mit dieser Tasache läßt es sich aber gut leben wenn man weiß, daß Tastaturbefehle in der Shell nicht wirken und man diese auch nicht mehr benutzen wird.

Da wir nun schon den Editor angesprochen haben, soll er hier auch ausführlich beschrieben werden. Es handelt sich dabei um einen voll in GEM integrierten Editor, mit maximal vier Arbeitsfenstern. Alle Fenster sind frei über den Bildschirm verschiebbar, ganz so wie man es auch von anderen GEM-Programmen gewohnt ist. Die Bedienung gestattet eine Steuerung über die Menüleisten, sowie über Tasten-Kommandos. Die Stärken des Editors sind dann auch diese Kommandos, die sich aus sogenannten 'einfachen Befehlen' zusammensetzen und auf den Funktionstasten gespeichert werden können. Ein Beispiel soll dies erklären:

Angenommen sie wollen den Editor mit der Taste F10 dazu veranlassen, drei Zeilen zu löschen und dann die Arbeit zu beenden. Sie tippen dazu folgendes Kommando:

FN10=/D;D;D;Q/

Wird anschließend die Taste F10 gedrückt, führt der Editor das Gewünschte aus. Jeder kann sich nun selbst ausmalen, was man mit diesen Funktionen alles anstellen kann. Natürlich lassen sich auch alle Einstellungen abspeichern, so daß sie nach einem Neustart wieder zur Verfügung stehen.

Im großen und ganzen läßt es sich mit dem Editor recht gut arbeiten. Die verfügbaren Funktionen sind ausreichend und arbeiten alle mit angemessener Geschwindigkeit. Einzig und allein das 'Scrolling' kann in Bezug auf die Geschwindigkeit nicht mithalten. Auch ein dadurch bedingtes störendes Nachlaufen des Cursors ist festzustellen. (In diesen Sachen bin ich wohl doch zu sehr von anderen Programmeditoren verwöhnt.)

Das RCS

Bei dem im System enthaltenen Resource Construction Set handelt es sich um das Programm von Kuma in der Version 1.2. Eine Beschreibung hierüber erübrigt sich, da schon ein Test in der Mai-Ausgabe der ST Computer erschien. Etwas darf aber nicht vergessen werden. Das RCS-Programm verträgt sich nicht mit den neuen Atari Computern der Mega-Serie. Man kann nur hoffen, daß dieser Fehler bald beseitigt wird, und das Programm allen Usern zur Verfügung steht.

Der Debugger

Der Debugger in dem System nennt sich DEBUG+ und ist ein symbolischer Debugger mit LINK-LOADER. Was ist ein Link Loader? Im Handbuch wird er wie folgt beschrieben: „Ein Link Loader ist ein Programm, das beim Einlesen der Objektmodule diese im Speicher des Rechners zusammen mit Libraryfunktionen zu einem ausführbaren Programm zusammenlinkt“. Es ist eigentlich eine feine Sache, denn man kann nun seinen Source compilieren und dann ohne Linkvorgang sofort das Objektmodul im Debugger starten. Diese Tatsache verkürzt die Programmentwicklung drastisch, da die langen Linkzeiten entfallen. Ansonsten kennt der Debugger alle Befehle, die zum Aufspüren der Bugs nötig sind. Alle Befehle hier aufzuführen ginge zu weit. Um an die Symbolnamen zu kommen, die man im Sourcecode vergeben hat, muß man allerdings den Linker darüber in Kenntnis setzen, daß man mit diesen arbeiten möchte. Dies geschieht über die Direktive '– debug', die in die Linker-Commandline zu schreiben ist.

Eine weitere Zugabe zum Paket besteht aus einem Make Utility, kurz MAKE genannt. All diejenigen, die schon mal mit dem UNIX Betriebssystem gearbeitet haben, werden sich an dieses nützliche Utility, zur Entwicklung von Programmen, das aus mehreren Object-Modulen besteht, sicher erinnern. Für alle anderen sei hier nochmals auf die Funktionsweise eines solchen Programms hingewiesen.

Bei der Entwicklung eines großen Programms teilt man den Sourcecode in einzelne Teile auf, auch Module genannt. Man tut dies aus mehreren Gründen. Ich möchte hier nur zwei davon nennen. Zum einem kann man Funktionen, die zusammengehören in einem Modul halten, z. B. alle Funktionen zur Ein-Ausgabe oder alle Grafik Funktionen. Dies erhöht den Überblick über das Programm. Man stelle sich nur vor, man suche in einem Sourcecode von über 1 MByte Umfang, eine Funktion, die z. B. einen Kreis zeichnet. Hat man sich das Programm in Module unterteilt führt eine Suche im Grafik Modul (wahrscheinlich) zum Erfolg. Der zweite wichtige Punkt besteht darin, die Compiler-Zeiten zu verkürzen. Fertigestellte und getestete Module müssen nur einmal compiliert werden bevor sie – mittels Linker – zum fertigen Programm zusammengebunden werden.

Die Aufgabe eines Make Utility besteht nun darin zu überwachen, welche Module geändert wurden und nur die geänderten neu zu compilieren, um danach wieder alles zusammenzulinken. Das Make Programm überwacht dies mit Hilfe der Datums- und Zeitangaben, die mit den einzelnen Files auf Disk geschrieben werden. Dies setzt natürlich eine immer richtig eingestellte Systemuhr voraus. Um ein einwandfreies Funktionieren zu gewährleisten, müssen in einem sogenannten make-file die Abhängigkeiten von Source und Object Modulen untereinander definiert werden. Im Handbuch ist hierüber ein großer Abschnitt mit vielen Beispielen vorgesehen, der die Handhabung der Make sehr detailliert behandelt.

Fazit

Das LATTICE C-Entwicklungssystem stellt eine professionelle Implementation der Sprache C auf dem Atari ST dar, der damit mehr denn je das Klischee eines Homecomputers verliert. Die Verfügbarkeit des LATTICE C Compilers auf anderen Computer-Systemen ist wirklich von Vorteil für all diejenigen, die sich gelegentlich mit Portabilitätsproblemen herumschlagen müssen.

Alle nötigen Hilfsmittel (vom Assembler einmal abgesehen) sind enthalten, die Funktionen des Atari Betriebssystem sind außerdem leicht erreichbar. Einer Entwicklung professioneller Software steht also nichts mehr im Wege. Was mir besonders gefiel, waren der sehr umfangreiche Compiler mit allen Libraries, der leistungsfähige Debugger sowie das ausgezeichnete Handbuch in deutscher Sprache, das sich stark an die englische Version anlehnt, aber tadellos übersetzt wurde und sehr übersichtlich aufgebaut ist. Einzig und allein die Tatsache, daß der Leser aufgrund des dünnen Papiers schon auf der Vorderseite erahnen kann was auf der Rückseite auf ihn zukommt, schmälerte bei mir die Begeisterung, die aufkam als ich das Buch in den Händen hielt. Dies mag sicher eine Preisfrage sein und hier haben sich Metacomco und der Vertreiber, es handelt sich um G. Knupe GmbH + Co KG Dortmund, was sehr erfreuliches einfallen lassen, kostet doch das gesamte Paket nur DM 298,– und ist nicht zuletzt auch dadurch eine ernsthafte Konkurrenz zu bereits erhältlichen Compiler Paketen.

(HJR)

G.Knupe GmbH + Co KG
Postfach 354
4600 Dortmund 1
Tel.: 02 31/52 75 31-32

CHRON DAS "PROFIPAKET"

- Eingabe und Simulation digitaler Schaltungen
- Worst-Case Simulation
- Schnittstelle zu HABACAD
- umfangreiche Bauteil-Bibliothek
- Anlegen eigener Bibliotheken
- komfortable Druckertreiber
- komfortable Plottertreiber

DM 1.898,00

TAPE-STREAMER

TAPE-20	20 MB	DM 1.998,00
TAPE-60	60 MB	DM 2.498,00

FESTPLATTEN

DISK-20	20 MB	DM 1.567,00
DISK-40	40 MB	DM 2.798,00
DISK-70	70 MB	DM 4.498,00

alle Geräte mit SASI-Interface

Lieferung per Nachnahme ★ bei Vorkasse Frachtfrei innerhalb 1-3 Tagen
Preise inkl. Mwst. ★ Gute Händlerkonditionen ★ Sofort ab Lager lieferbar

veba-applications

Inh. Victoria Becker ★ Breslauer Str. 12 ★ D-8034 Germering

ATARI ST

steckbar steckbar Speichererweiterungen

auf **1 MByte** 260/520 STM **275,–**
auf **2,5 MB / 4 MB** a.A.
260 ST, 520 STM, 1040 STF

Jede Erweiterung einzeln im Rechner getestet! Sehr einfacher Einbau. **Ohne Löten einbaubar!!!** Gut bebilderte, ausführliche Einbauanleitung. Vergoldete Mikro – Steckkontakte für MMU-Adapter. Optimale **Schonung des MMU-Sockels** Achten Sie auf **Mikro-Steckkontakte** Kein Bildschirmflimmern!!! Ohne zus. Software, ohne zus. Stromversorgung. Test in ST 4/86.

ECHTZEITUHR 129,–

Jede Uhr im Rechner getestet! Einbau durch Einstecken in den Rechner. Freier ROM Port – für spätere Erweiterungen. **Uhrzeit, Datum, Wochentag Anzeige.** Dank Lithium-Batterie ca. **10 Jahre Laufzeit. Hohe Genauigkeit.** Schaltjahrerkennung.

Stets günstig
Drucker und Monitore
Komplettsysteme preisgünstig!

Händleranfragen erwünscht.

WEIDE-ELEKTRONIK

Regerstr. 34 · 4010 Hilden · Tel.: 0 21 03/4 12 26
Kostenloses Info!!!

Schweiz	Niederlande	Belgien
SENN Computer AG	Commedia	MICRO-CONNECTION
Langstr. 31	1e Looiersdwarsstr. 12	St Katelijnevest 18
CH-8021 Zürich	1016 VM Amsterdam	2000 Antwerpen
Tel. 01/241 73 73	Tel. 020/23 17 40	Tel.: 03/231 15 40

GFA BASIC
NEU auf Modul

NLQ NLQ NLQ

- Aufrüstsatz für alle
Epson **MX, RX, FX, JX** Drucker
Apple Macintosh Drucker
Emulation für (FX & JX)
Viele Features! Info anfordern!
FX **199,–** MX **179,–** RX 80 **149,–**

GFA-BASIC V2.0 MODUL 198,–

3,5" Laufwerke

Disketteneinzelstation (1 MByte)	**398,–**
Diskettendoppelstation (2 MByte)	**698,–**
im PC Gehäuse 2 Laufwerke á 1 MB	**749,–**

CO-Prozessor 68881 890,–

inkl. Software für Megamax C, Mark Williams C, DRI C, Prospero Fortran 77, Modula II, CCD Pascal ⊕
erhöht die Rechengeschwindigkeit z.T. um Faktor 750: einfachster Einbau, rein steckbar - ohne Löten!
unbedingt **ausführliches** INFO anfordern!

Trackball statt Maus (platzsparend u. schnell) **119,–**

---

dBMAN™

dBMAN-MANAGER* 198,- DM
Relationale Datenbank unter GEM — eigener Editor — Funktionstastenprog. — TXT, DIF, SDF, Delimited Formate — Schnittstelle zu GEM-Programmen — Report- und Listenausgabe

dBMAN-LITERATUR* 189,- DM
Unter GEM — Schnittstelle zur WORDPLUS — Druckformat frei erstellbar — Felder frei programmierbar — Hilfstexte — 6 Indexfelder

dBMAN-Adressen* 99,-
Unter GEM — Serienbriefe — Druckformate frei erstellbar — Felder frei programmierbar — Hilfstexte — Report — ACCESSORIES voll verfügbar — uvm.

* RUN-ONLYVERSIONEN natürlich in Deutsch

dBMAN-GEMTOOL 199,-
Source-Code mit: Serienbriefschnittstelle, PULL-DOWN-MENÜ, MAUS-MENÜ, Maskeneditor, Suchfunktionen, GEM-Schnittstelle, Dateiauswahlformular, Memorydateien

dBMAN-GEMSCHNITTSTELLE 69,- DM
GEM-Programme aus dBMAN aufrufen.

SCHRÖTTLE-SHELL Ver. 4.2 DM 98,-
DIE UNIX-BOURNE-ähnliche Shell als ACCESSORIE.

CHRON Basispaket 248,- DM
Schaltungseingabe + Logiksimulation unter GEM

Nur per Nachnahme oder Euroscheck zuzüglich 4,- DM Versandnebenkosten bei:

COMPUTER-MAI Weißenburger Platz 1
8000 München 80
Tel. 089 / 448 06 91 14.00 bis 17.00
Händleranfragen erwünscht.

DIE GEM PROFIS KOMMEN

Public-Domain Service

Die Neuheiten dieses Monats

In diesem Monat, so glauben wir, sind wieder einige interessante Programme bei uns eingetroffen. Die Hauptthemen liegen diesmal in den Bereichen Schulungssoftware, Utilities, Druckertreiber, Adressverwaltung und Spiele. Um nur einige zu nennen: Ein 3D-Sternenhimmel, ein Programm mit allen Fragen der theoretischen Führerscheinprüfung, eine resetfeste Ramdisk für das neue TOS, Shells für Pascal und Signum, Hardcopytreiber für NEC P6, Epson LQ800, u. a. und gute Adressverwaltungen. Im Bereich Spiele erhielten wir eine Version des Altschlagers PACMAN und Diamond Miner mit frei editierbaren Feldern. Wir hoffen, daß Sie mit den Programmen Freude haben werden und bedanken uns im Namen aller Benutzer bei den Programmautoren.

Ihre ST-Computer Redaktion

86

● *Carpet:* Berechnungsprogramm von 3D-Funktionen. Mit Hidden-Line Algorithmus und mit Beleuchtungseffekten. Funktionseingabe und frei einstellbare Parameter. (s/w) ● *3D-Plot:* Ähnliches Programm in compiliertem GFA-Basic. Obwohl die Programme ähnlich sind, so bietet doch jedes seine Spezialfunktionen. (s/w) ● *Funktionsplot:* Funktionseingabe im Programm, verschiebbare Achsen, mehrere Darstellungsarten (s/w)

87

● *Sternbild:* Zeigt den Sternenhimmel, oder bestimmte Ausschnitte zu verschiedenen Zeiten. Planetennamen auf Anklicken. S7W und Farbe). Im Farbbetrieb erzeugt es auch ein räumliches Rot-Grünbild

88

Druckertreiber

● *NEC-EMU:* Der bereits in den NEWS angekündigte Hardcopytreiber für die NEC Px-Reihe. Erzeugt Hardcopies in 12 verschiedenen Größen und Kontrasten. Enthält Druckerspooler ● *LQ 800:* Ein Hardcopytreiber für den EPSON 24 Nadeldrucker in drei verschiedene Größen ● *NEC_Code:* Ein Programm zur Einstellung des NEC P6 /P7 u. ä. ● *SIMPLE:* Komfortables Hardcopy-Programm mit vielen Extras ● *STX-80:* Hardcopytreiber für diesen preiswerten Thermodrucker ● *WORDPLUS:* Druckertreibersammlung für folgende Drucker: NL10, FX1000, SP1000I ● *1st prop:* 1st Proportional Treiber für folgende Drucker: HR15, FX85.

89

● *Führerschein:* Haben Sie schon den Führerschein? Dann haben Sie Glück. Falls nicht hilft Ihnen dieses Programm zumindest bei der Theorie. Ca. 500 Fragen sind enthalten ● *Bundesliga:* Zwar hat die Saison schon begonnen, aber entschieden ist ja wohl noch nichts. Diese Programme dienen zum Anlegen einer Tabelle, editieren aller Spiele von der 1., 2. Bundesliga, bis hin zur untersten Bezirksliga.

90

Utilities

● *TOPS:* The Other Pascal Shell Erlaubt die Einbindung von Tempus, erzeugt Crossreferenz und enthält Befehle zum Kopieren, Löschen und Umbenennen von Files ● *SIGNUMShell:* Eine Shell zu Signum. Man kann nun per Menü zwischen den Signum Editor, den verschiedenen Druckprogrammen, den Zeicheneditoren hin und herschalten ● *Tempelmon 1.8:* Neue Version des weitverbreiteten Monitors. Läuft nun auch in Farbe ● *Maxidisk 4Mb:* Die komprimierende Maxidisk mit max. 4MByte Kapazität. Läuft auch auf Blittertos. Komprimiert automatisch ihren Inhalt. Resetfest ● *RAMDISK:* Gleiches Programm wie MAXIDISK, nur ohne Komprimieralgorithmus. Beide Ramdisk enthalten ein Autocopyprogramm zum kopieren bestimmter Files in Ramdisk ● *RCS.Wandel:* Wandelt ★.h Files nach Basic und Assembler.

91

Lernprogramme

● **Pauk Lerndatei mit komfortabler Steuerung, eigener Dateneingabe, wiederholung falscher Fragen nach Lernkonzept. (s/w)** ● *Hypervoc:* Sehr schön gestaltetes Vokabellernprogramm. (s/w) ● *ST-Dic:* Fremdsprachenlernprogramm mit schon vorhandener großer Lateinbibliothek ● *QUIZ:* Eine Art Trivial Persuite. Enthält Fragen mit drei möglichen aber nur einer richtigen Antwort. Eigene Fragen einbinden möglich. Enthält (ATARI ST-Einsteigerkurs).

HYPERVOC

92

Utilitydisk

● *Superbatch:* Batch-Prozessor: bedingte Batchbearbeitung (if) / Batch Protokoll auf Bildschirm, Drucker, Datei / Verzweigung, Fehlerbehandlung ● *ST-Klick:* Multifunktionsaccessoir: Notizen, Wecker, Drucker init, Schreibmaschine, Kalender, Taschenrechner (s/w) ● *XREF:* Crossreferenz zu C ● *Term:* Terminalprogramm mit Xmodem ● *Altaformat:* flexibles Formatierprogramm (9 – 11 Sektoren) (bis 86 Tracks).

93

ADR_2 Semiprofessionelles Adressverwaltungsprogramm mit Serienbrieferstellung (mit und ohne 1st Word). Eigene editierbare Druckeranpassung (Steuerzeichen, Zeichenwandlung). Erzeugt Adress- Telefon und Geburtstagslisten. Druckt Labels, Etiketten nach freidefinierbarem Format. (s/w) ●*Dat_Text* Adressverwaltung für Mehrpersonenzugriff auf eine Datei. Datenzugriff nur durch Passwort (s/w)

94

Spieledisk

● *MacPan:* Eine schnelle Pacman-Version für den ST. Per Joystick ins Labyrinth auf der Suche nach Powerpills (s/w) ● *Solitär:* Dreidimensionales Solitär mit sehr ansprechender Grafik (s/w) ● *Galaxy Sprengmeister:* Brettspiele zur Förderung der Gehirnaktivität (s/w)

95

Spieledisk

● *Diamond Miner:* Eine Spiel ähnlich Boulder Dash. Sehr schnelle Grafik und 30 Level. Eingebauter Feldeditor, bei dem nach Belieben eigene Spielfelder generiert werden können. (s/w) ● *Snake:* Zwei Schlangen versuchen sich den Weg abzuschneiden. (s/w) ● *Minenfeld:* Ein wagemutiger Schatzsucher begibt sich durch verschiedene Minenfelder und muß versuchen die Minen zu umgehen. (s/w) ● *Missle:* Missleattack (s/w)

Public-Domain Service

Die Diskette 1...20 sind auch weiterhin erhältlich. Schauen sie dazu bitte in einer der vorherigen Ausgaben nach oder fordern Sie die Liste an (frank. Rückumschlag).

21. Diskette
● *Adress*: Private Adressverwaltung ● *Etikett*: Etikettendruck ● *Haushalt*: Privatkontenverwaltung ● *Selektor*: Dateiverwaltung (30 frei erstellbare Felder)

22. Diskette
● *Plot*: Grafische Darstellung von Wertepaaren ● *Myterm*: Terminalprogramm (GfA-Basic] ● *VKT*: Vokabeltrainer

23. Diskette
● *DGDB*: 'Das große deutsche Ballerspiel', Aktionspiel siehe RELAX 12/86 [f] ● *Kalah*: Das bekannte Brettspiel ● *Fractals*: Schnelle 'Apfelmännchen-Version' in Assembler.

24. Diskette
● *Taurus*: Strategiespiel für mehrere Spieler. Siehe Relax 12/86. Schöne Grafik, viele Optionen [f], (leider nur Mega-Rechner)

25. Diskette
● *Disk Engineer*: Diskutility (Rescratch, ...] ● *Omikron Basic*: Run-Only Interpreter

26. Diskette
● *Codetabelle*: ASCII-Tabelle (Accessoir) [s/w] ● *Life_Game*: Das bekannten Generationenspiel (Accessoir) [s/w] ● *Term*: Und noch ein Terminalprogramm ● *Biorhythmus*: ST-Basic schlägt zurück [s/w] ● *Konto*: Kontoführung in GfA-Basic [s/w] ● *KX P1091*: Accessoir zur Druckersteuerung

27. Diskette
● *NEC-Treiber*: Hardcopy-Treiber für 24-Nadeldrucker (NEC und kompatible). Aufruf über ALT + HELP ● *ST-Sound Demo*: Die Siegerdemo des ST-Computer-Sound-Wettbewerbs. 'Zottl, Knurp und Schnatter hauen rein' (siehe Titelbild ST-Computer Nr. 12)

28. Diskette
● *GfA Basic Programme incl. Source* ● *Funkplot* [s/w] ● *Terminal*: Terminalprogramm in GfA [s/w] ● *Konverter*: Konvertiert ST-Basic in GfA-Basic ● *ST-Kill*: Utilityprogramme ● *Sprite-Editor*: [s/w]

29. Diskette
● *Diskutility*: Der Weg ans Directory (manipuliert Löschschutz, erzeugt waschechtes IBM-Format, und mehr) [incl. Source in Pascal] ● *Adress_pas*: Adressverwaltung. ● *ST-Graph*: 3-D Balkengrafik [s/w] [incl. Source in GfA]

30. Diskette: Spielesammlung
● *Bohnen* ● *Schiffe*: Schiffe versenken ● *Mastermind*: Superhirn für Superhirne [f] ● *Dragon_hunt*: Ein äußerst gefräßiger Drachen will ausgerechnet Ihnen an den Kragen [f] ● *Memo*: Memory (2 Versionen) ● *Solitaire*: Das bekannte 'Taschenspiel' ● *4 Gewinnt*: Spiel in GfA-Basic incl. Source [s/w]

31. Diskette: Spielesammlung
● *Patience*: Wunderschöne Version dieses Kartenspiels (siehe RELAX Nr. 1/87 [s/w] ● *3D-Irrgarten*: Finden Sie einen Weg aus einem dreidimensionalen Labyrinth. Trotz ST-Basic sehr schnell [s/w] ● *Avalun*: Sie verwalten ein Königreich (ST-Basic)

32. Diskette
● *Mathlib*: Eine umfangreiche Mathematikbibliothek für Pascal. Trigonometrische, Matrizenalgebra und vieles mehr, direkt zum Einbinden ● *Finanzmathe*: Berechnung von Zinsen, Laufzeiten, Laufzeit, Tilgungsplänen, etc.

33. Diskette: diverse Utilities
● *Encrypt*: Ver- und Entschlüsseln von Dateien ● *Ramdisk*: Ramdisk beliebiger Größe ● *Recover*: Regeneriert gelöschte Files (!!) ● *Storm*: VT52 Terminal + Source in Assembler ● *Freeram*: Zeigt freien Speicher

34. Diskette: Der Assembler zum Kurs
● *Assemblerpaket nach M 68000-Standard* incl. 2-Pass Assembler, Disassembler, Debugger, Editor.

35. Diskette
● *Andromeda*: Zeichenprogramm (DEGAS-ähnlich) ZOOM, Beschriften in vier Richtungen und allen Größen, Kopieren, Linienstil / Art / Breite und alle normalen Grafikbefehle [s/w] ● *3D-Konstruktion*: Zum Entwurf dreidimensionaler Gegenstände [s/w] ● *Globus*: Animierte Erdballgrafik

36. Diskette
● *CLI*: vollständiger Kommandointerpreter (CP/M, MS-DOS Oberfläche) ohne Maus. Ausführen von Batch-Files, Autostart, Kopieren auf Ramdisk, Belegung der Funktionstasten, Ausführen von Programmen (TOS + GEM), etc. ● *IBM Drive*: Ermöglicht den Filetransfer IBM < - > ST ● *Drupatch*: Zum Ausdruck von Source Files. Mit Zeichenwandlung wie 1ST Word, universell anpaßbar. ● *DB_Etikett*: Druckt Etiketten aus DB-Master

37. Diskette
● *SCIENTFIC*: Zeichensatzeditor. Erzeugt Anpassungsfile für 1st Word [s/w] ● *Convert*: Konvertiert ST- und Microsoft Basic Programme in GfA-Basic ● *Check*: Rechtschreibkorrekturprogramm. Erstellt eigenes Lexikon, Übernahme der neuen Wörter, Korrektur direkt im Text.

38. Diskette
Alle Bilder des ST-Kalenderwettbewerbs, die in Heft 1/87 veröffentlicht wurden, incl. Picture-Compactor.

39. Diskette
● *Jackpot*: Einarmiger Bandit, incl. Sourcecode in GFA-Basic [s/w] ● *Lucky*: Spielautomat [s/w] ● *Hanoi*: Türme von Hanoi.

40. Diskette
● *Deluxe Fontmaster ST*: Druckprogramm zum Erstellen von Karten und Tabellen, mit vielen Zeichensätzen, Proportionalschrift und Blocksatz, eigener Text- und Zeichensatzeditor. Bis zu 10 Schriftarten in einem Text mischen und vieles mehr [s/w]

41. Diskette
● *Painter*: Zeichenprogramm unter GEM. Verfügt über einige Besonderheiten. Z. B. Zoom, Linien und Füllmustereditor, Blockoperationen (Verschieben, Drehen,...) [s/w] ● *VIKA*: Bedrucken von Kassetten und Videohüllen [s/w] ● *Kalender*: Diverse Kalenderberechnungen. U. A. Ausgabe eines Jahreskalenders auf Drucker oder Bildschirm, mit Anzeige aller Feiertage (sehr wichtig)

42. Diskette
● *FRAM 77*: Statische Berechnungen für Konstrukteure. Sehr umfangreich ● *Meßwert*: Grafische Darstellung von Meßwerten [s/w], incl. Sourcecode in Pascal

43. Diskette
● *System*: Accessoir ähnlich dem Kontrollfeld. Besonderheiten: Uhr und 4 Weckzeiten, Druckereinstellung (Rand, Schriftart,...) [s/w], incl. Sourcecode in Pascal ● *Snaphelp*: Hiermit kann SNAPSHOT (1st Word Utility) auch auf DEGAS angewendet werden, incl. Sourcecode in Pascal

44. Diskette
Spieledisk
● *TTT*: Dreidimensionales Tic Tac Toe (4 * 4 * 4) [s/w] ● *El Bozo*: El Bozos Window Puzzle ● *Senso*: Fordern Sie Ihr Gedächtnis [s/w] ● *Kalah* ● *Tri Hex*: Räumliches Tic Tac Toe (3 * 3 * 3) [s/w] ● *Börse*: Spekulieren am Börsenmarkt [s/w] ● *Solitär*: [s/w]

45. Diskette
● *Complex*: Bibliothek f. komplexe Berechnung ● *Master*: Generationsberechnung mit frei definierbaren Regeln [s/w] ● *Pegasus*: Malprogramm [s/w] ● *Con_ST*: Terminalprogramm u. a.

46. Diskette
● *Termin*: Terminplanung als Accessoir [s/w] ● *KFZ-Stat*: Statistikprogramm für's Auto [s/w] ● *Lager*: Lagerverwaltung [s/w] ● *ST_Phone*: Telefonkostenberechnung [s/w]

47. Diskette
Utilitydisk
● *M_Copy_87*: Backupprogramm mit vielen Optionen und Musik ● *Copy_FMT*: Backupprogramm als Accessoir ● *Uhrzeit*: diverse Utilities zur eingebauten Uhr ● *1st_step*: Kalender und Biorhythmus [s/w] ● *Lotto*: [s/w] ● *P2_Init* u.a.

48. Diskette
● *Word 1.4*: Programm zum Erstellen eines luxuriösen Lexikons für 1st Wordplus

49. 50. 51. Diskette
● *Volksforth 83*: Die neueste Version des Forthinterpreters. Die Software zum Forthkurs. Alle drei Disketten gehören zum Forth-System.

52. Diskette
● *Molekül*: Darstellung chemischer Moleküle. In s/w und Farbe, wobei in Farbe mit 3D-Effekt. Atom, Ionen oder Van der Waalsradius. Ein Molekül kann gedreht und sogar animiert werden.

53. Diskette
● *Matrix*: Komfortables Matritzenberechnungsprogramm mit voller GEM-Steuerung (Icons, Windows, Menüzeile). Maximal 16 * 16 Matritzen, doppelte Genauigkeit (64 Bit). ● *LIB*: Sammlung von ST-Pascal-Bibliotheken. Incl. Source und Anleitung. Stringumwandlung (str < - > int, long_int, real) Datums + Uhrzeit-Rechnungen Textformatierungen (links, rechtsbündig, zentriert, u.v.m.)

54. Diskette
Spielesammlung
● *Bounce*: Bouncing Bubbles. Schnelles Actionspiel für Joystick und Tastatur (s/w) ● *Domino*: Gefräßige Würmer auf Nahrungssuche (s/w) ● *Trio*: Verbessertes Tic-Tac-Toe mit Verschieben (s/w) ● *Disc*: Die Serie der Spielautomaten reißt nicht ab (s/w)

55. Diskette
Editorendisk
● *TOSEDIT*: Editor unter TOS. Gleichzeitiges Bearbeiten von zwei unabhängigen Texten. Reichhaltige Befehle ● *EDIMAX*: Editor unter GEM. Mit Funktionstastenbelegung. Weiterhin zeichnen sich EDIMAX durch sehr schnelles Scrolling aus.

56. Diskette
● *Ausland*: Fremdsprechentrainer Englisch mit Sprachschatz für Fortgeschrittene. ● *Fugger*: Verwalten Sie ein Fürstentum und versorgen Sie ihr Volk. (s/w) ● *Yahtzee*: Altbekanntes Zockerspiel auf den ST umgesetzt.

57. Diskette
Wissenschaftliche Anwendungen
● *Kepler*: Grafische Darstellung von Himmelskörper-Bewegungen nach den Keplerschen Gesetzen. Variation von Masse, Radius, Zeit, potentieller und kinetischer Energie. ● *Kerne*: Der Krieg der Kerne, eine futuristische Schlacht; im Innern eines Computers suchen Rechenprogramme sich gegenseitig zu zerstören. ● *Meßwert*: Verarbeitung von Meßergebnissen und grafische Darstellung (s/w).

58. Diskette
● *SM-Konverter*: Konvertiert SM-Manager-Texte in 1st Word (Wordplus)-Texte und natürlich auch zurück. ● *Compress*: Komprimiert beliebige Files. ● *Label*: Ändern des Diskettennamens. ● *Eti-Master*: Etikettendruck zu DB-Master ● *Urlaub*: Hilft den Urlaub zeitoptimiert zu berechnen.

59. Diskette
● *Themadat*: Assoziative Datenbank mit bis zu 200 Datensätzen.

60. Diskette
● *Synthi*: Erzeugt div. Sounds, die abgespeichert und über 'DOSOUND' in eigene Programme übernommen werden können (incl. der Source-Code) (s/w) ● *Waschi*: Waschhilfsprogramm für Junggesellen (s/w) ● *Expert*: Ein einfaches, lernbegieriges Expertensystem ● *Adressen*: Adressverwaltungsprogramm mit Etikettendruck und einfacher Textverarbeitung

61. Diskette
● *Hecomini*: Erzeugt Hardcopies in Miniaturformat (48 mm * 68 mm) (s/w) ● *Helper*: Hilfsprogramm bestehend aus Diskmonitor und Druckerutility ● *Bidipo*: Scrollt eine Bilderkette über den Schirm. Ideal für eigene Diashows oder Werbegrafiken ● *Datobert*: Grafische Darstellung. Linien- und Kuchengrafik, sowie in dreidimensionaler Balken- und Flächengrafik. Überlagerung einzelner Datenserien (s/w) ● *Direct all*: Directorydruck ● *Überweisung*: Überweisungsdruckprogramm aus ST 6/87.

62. Diskette
Midi-Diskette
● *MIDIDRUM*: Rhythmusmaschine mit leicht und vielseitig editierbaren Sounds ● *MIDIPLAY*: Erlaubt das Einspielen, Speichern und Ausgeben von Midi-Sequenzen, incl. Source-Code in GFA-Basic.

63. Diskette
Spielesammlung
● *Fuffy*: Sehr schön gestaltetes Kniffel-Spiel mit zusätzlicher Auswertegrafik (s/w) ● *Snake*: Eine Schlange schlängelt sich durchs Leben, auf der Suche nach Fressen. Man achte auf Hindernisse (s/w) ● *Bomber*: Scramble-Spiel (f) ● *Reversi*: Der Name sagt alles (f).

64. Diskette
● *E-PLAN*: Konstruktionsprogramm für elektronische Schaltungen. Alle elektronischen Bauelemente sind vorhanden. Schaltung können auch beschriftet und die Bauteile in die Stückliste übernommen werden.

65. Diskette
Mal- und Zeichenprogramme
● *Columbia*: Malprogramm mit vielen Extras. Mehrere Bildschirme, Blockoperationen, Spiegeln, Verzerren, Drehen, Lupe (s/w) ● *Paintlux*: Malprogramm mit zusätzlicher Bibliotheksverwaltung. Dadurch können z. B. elektronische Bauteile aus der Bibliothek in das Bild eingefügt werden (s/w).

66. Diskette
Spieledisk
incl. Source-Code in OMIKRON Basic
● *MAZIACS*: Abenteuerspiel. Unser goldsuchender Held im Labyrinth, im Kampf mit bösartigen Spinnen und anderen Hindernissen (s/w) ● *Quadrat*: Bilderpuzzle. Ordnung im Chaos (s/w).

Diskette 67
Utilities
● *Diskchecker*: Prüft eine Diskette auf Fehler. Formatier- sowie Schreib- und Leseprüfung (s/w). ● *DEGAS ELITE Fontmaker*: Erzeugt das DEGAS ELITE-File Assign.SYS. Damit wird die Einbindung neuer Fonts stark vereinfacht. ● *PRINTSET*: Druckeranpassungshilfe. Sonderzeichenausgabe. ● *TESTBILD*: Das Farbmonitor-Testbild. ● *ICONEDIT*: Sprite-, Maus- und Füllmuster. Übernahme in eigene Programme.

Diskette 68
Mal- bzw. Zeichenprogramme
● *Studio*: Zeichnen in DIN A4-Format. Blockverwaltung. Automatische Bemaßung. Ausschnitts-Hardcopy. ● *Van Gogh*: Mal- und Animationsprogramm. 64 bewegte Blökke. Incl. Blockeditor (f).

Diskette 69
● *Astrolabium*: Zeichnen von Sternkarten- oder Himmelsausschnitten, zu beliebiger Zeit, an beliebigem Ort. Einstellbare Blickrichtung, -höhe und -winkel. Incl. Daten von ca. 9000 Sternen (s/w).

Diskette 70
● *Orbit*: Zeigt die Laufbahn von Satelliten als bewegte Grafik Über der Weltkarte. Z. B. für Funkamateure (s/w). ● *High End*: Programm zum Berechnen von Lautsprecherboxen aller Art. Lautsprecher und Boxenmaße. Datei für Speaker enthalten (s/w).

Diskette 71
Spieledisk
● *Ballerburg*: Zwei verfeindete Burgherren zerstören sich gegenseitig ihre Burgen. Steuerung der Kanonenwinkel und der Pulvermenge (s/w). ● *Stagger*: Reversiumsetzung (s/w). ● *Hotelier*: Hotelverwaltungsspiel. Leiten Sie Ihr Hotel und kämpfen Sie gegen die Konkurrenz (s/w). ● *Roulette*: Was mag dies wohl sein? (s/w)

Diskette 72
● *ZREF*: Crossreferenz für C-Programme. Erzeugt Variablenliste mit Funktionsaufrufen, Typen, defines, structs, Zeilennummern und vielen anderen Optionen. ● *Plotter*: Werteplotter für wissenschaftliche Anwendungen. Lineare-, Lagrange-, 3-Punkte-Interpolation sowie Regressionsanalyse. Nachträgliches Editieren (s/w). ● *Nicelist*: Ausdruck von Source-Codes mit Seitennummern, Dateiname und Datum. Sonderzeichenanpassung.

Diskette 73
● *Videoarchiv*: Videokassettenverwaltung. Leicht bedienbar. Mit Etikettendruck (s/w). ● *Lohnsteuer*: Lohnsteuerberechnungsprogramm. ● *Heimmanager*: Kontrolle über die private Haushaltsabrechnung. Jahresübersicht (s/w).

Diskette 74
Ramdiskpaket u.a.
● *MAXIDISK*: DIE Ramdisk. Resetfest (Inhalt nach Reset noch vorhanden). Komprimiert ihren Inhalt (eine 500K-Ramdisk kann wesentlich mehr aufnehmen als 500K). Frei wählbare Größe, frei wählbare Laufwerkskennung. ● *COPY*: MAXIDISK-Utility. Kopiert Inhalt eines Ordners beim Systemstart in die Ramdisk. ● *TIME*: MAXIDISK-Utility zum Stellen des Datums und Uhrzeit. ● *Mono Neo*: Speichert bei Alternate + Help den Bildschirm auf Diskette. ● *Autocopy*: Ähnlich dem Programm COPY. Kopiert bestimmte Dateien in beliebige Ordner. ● *Datebook*: Erinnert beim Systemstart an wichtige Termine oder Geburtstage. u.a.

Diskette 75
Fontdiskette zu Fontmaster (PD40): Neue bzw. erweiterte Fonts und Rahmen. Ergänzung zu Fontmaster (PD40).

(s/w): nur monochrom (f): nur Farbe

Diskette 76
Schulungssoftware
● *Periode*: Das Periodensystem der Elemente. Chemielernprogramm mit vielen Optionen. Enthält die Daten jedes chemischen Elements (s/w). ● *Statistik*: Grafische Darstellung und Datenauswertung (s/w).

Diskette 77
Utilitydisk
● *FCOPY 2.0*: Schnelles Kopierprogramm. Formatieren, Verify, Multiple Copy, Scan und Teilkopie. ● *Speeder*: Verdoppelt Schreib- und Lesegeschwindigkeit. ● *Filecopy*: Kopieren von Einzeldateien. Auch mit einer Floppy. ● *Formater*: Erhöht die Diskkapazität. ● *Fastformat*: Formatiert Schnelladedisks. Durch besondere Anordnung der Sektoren wird immer schneller geladen. ● *Taskcopy*: *Diskbackup im Multitasking* ● *Profilramdisk*: *Resetfeste Ramdisk mit Ausführungsprofil*

Diskette 78
Utilitydisk
● *AESLIB*: Bibliothek für GFA-Basic. Enthält komplette AES-Bibliothek. Mit Linker, der jede benötigte Prozedure automatisch zum Programm hinzulädt. ● *LINE_A*: Line_A Binding für C-Programme. ● *Conv.WP*: Konvertiert ASCII-Dateien in den WP-Code von 1st Word. Dadurch Umbruch, Formatieren etc. möglich. ● *Player*: Programm zum Abspielen von Interruptmusik incl. Source in C. ● *Musik*: Ähnliches Programm aber als Accessoir. ● *Setz_Uhr*: Programm zum Stellen der Zeit + Datum incl. Source in Assembler.

Diskette 79
Etikettendruck
● *Label Expert*: Druckt alle Art von Etiketten. 3,5 Zoll- und 5,25 Zoll-Disketten, Audio-Kassetten, Video-Kassetten, Ordner u.a. (s/w). ● *Grafikett*: Bedruckt Disklabels auch mit Grafik, die aus beliebigem Grafikbild ausgeschnitten wird (s/w). ● *Etikettdat*: Druckt Adressenaufkleber (s/w). ● *Etikette.GFA*: Luxuriöses Disklabeldrucker. Mit vordefinierter, artbezogener Grafikeinbindung (s/w).

Diskette 80
Spieledisk
● *U_Boot*: Ziel dieses Spiels ist es, U-Boote im Meer mit eigenen Sonden zu orten und ihnen auszuweichen (s/w). ● *Ärger*: Eine Umsetzung des wohl bekanntesten Brettspiels 'Mensch ärger dich nicht' auf den ST (s/w).

Diskette 81
Malprogramm
● *Public Painter*: Malprogramm mit interessanten Fähigkeiten. Verarbeitet z. B. gepacktes Format oder Bilder von SNAPSHOT. Eingebaute Lassofunktion zum Ausschneiden beliebiger Teilbereiche. Blockmanipulation ähnlich EASYDRAW. Vergrößern, Verkleinern, Drehen, Biegen, Verzerren, Überschneiden. Drucken von Bildausschnitten (s/w).

Diskette 82
Malprogramm
● *Masterpainter*: Läuft in allen drei Auflösungen. Konvertiert jede Auflösung in beliebige andere. Blockbearbeitung. Vergrößern, Verkleinern, Rotieren etc., bis zu 24 Screens. Ausschnittdruck. ● *Little Painter*: Malprogramm mit P6-Treiber. Rasteroperation, Musterspray, Verwischen, Drehen, Biegen, Blockladen und -speichern, Fontlader (s/w).

Diskette 83
● *Starofix*: Komfortabler Zeichsatzeditor für Down-Load Zeichen. Für Star NL10 und kompatible. Ein Zeichensatz wird per Autostartprogramm in den Drucker geladen. Ausdruck mit jedem Programm mit neuen Zeichen. Viele Fonts schon vorhanden. ● *Starlist*: Ausdruck von ASCII-Dateien mit Sonderzeichenwandlung.

Diskette 84
Druckedit: Zeicheneditoren für 1st Word-Druckeranpassung. ● *Draft Pin*: Die Zeichen werden per Maus editiert und dann automatisch in beliebiges .HEX-File eingebaut. Ideale Ergänzung zu 1st Word bzw. Wordplus (s/w).

Diskette 85
Spieledisk
● *Skat*: Eine Runde SKAT gefällig? Sie spielen gegen zwei vom ST gesteuerte Zocker. Ansprechende Grafik (s/w). ● *Delta*: Gehirnmuskeltraining. Ähnlich dem Zauberwürfel (f). ● *Grusel*: Grafik- und Sounddemo. In der Geisterstunde ist auf dem Friedhof allerhand los. Skelette, Äbte und Gespenster treiben ihr Unwesen (f).

Sonderdisks

Die folgenden Programme sind nicht Public-Domain.
Sie können aber bei uns bezogen werden.

A.) – *TOS*: Die letzte Disketten-Version des TOS vom 6.2.1986. Zum fehlerfreien Betrieb des Rechners benötigen Sie diese Version, da z. B. mit älteren Versionen manche Ordner nicht gelesen werden (0 Bytes free). Unkostenbeitrag 15,– DM incl. Diskette.

B.) – *RCS*: Das Resource Construction Set aus dem ATARI-Entwicklungspaket. Unverzichtbar bei der Erzeugung von Objektbäumen, Dialogboxen, Drop-Down Menüs. Kurzum: Ein Muß für GEM-Programmierung. Unkostenbeitrag 15,– DM incl. Diskette.

Bis auf die Höhe des Unkostenbeitrags gelten die gleichen Versandbedingungen wie bei der Public-Domain-Software.

Versandbedingungen:

Sämtliche Disketten können Sie direkt von der Redaktion erhalten. Wir haben für Sie den schnellstmöglichen Versandservice eingerichtet.
24h-Service – Lieferung innerhalb eines Werktages

1. Anruf genügt:
ST-Computer Redaktion
Tel.: 0 61 96 / 48 18 11
BRD / 61 96 / 48 18 11

2. *Schriftliche Bestellung*:
– Der Unkostenbeitrag für eine Diskette beträgt DM 10,–
– Bezahlung per Scheck oder Nachnahme (Nachnahme nur im Inland).
– bei Nachnahme zuzüglich DM 3,50 Nachnahmegebühr
– Fügen Sie dem Betrag folgende Versandkosten (Porto + Verpackung) bei:
Inland DM 5,– Ausland DM 10,–
– Legen Sie bitte, falls zur Hand, einen Aufkleber mit Ihrer Adresse bei.

Die Bezugsadresse lautet:

ST-Computer Redaktion
MERLIN-Computer GmbH
'PD-Service'
Postfach 59 69
D-6236 Eschborn

Bei den Disketten handelt es sich um FUJI MF1DD, die wir uneingeschränkt für ein- und zweiseitige Laufwerke empfehlen.

Bei Fragen bezüglich der Disketten wenden Sie sich bitte an die Redaktion.

ALICE aus dem Wunderland

ALICE Pascal ist ein PASCAL Interpreter. Es gibt ihn für den IBM-PC und den ATARI ST. Und er ist einzigartig. Lesen Sie warum.

Einführung

Angesichts der unübersichtlichen Fülle von Computersprachen, erhebt sich gerade für den Neuling die Frage, mit welcher Sprache er seine ersten Schritte im fremden Computerland unternehmen soll. Fragt man einen alten Hasen nach seiner Meinung, so erhält man bei n befragten Personen 2n+1 Antworten, je nach Einstellung und Vorbildung des Befragten. Eines steht aber fest: Das serienmäßig mit dem ST ausgelieferte BASIC ist so schlecht, daß man besser gleich einen anderen Interpreter (oder Compiler) mitkauft. Und wenn man seinem Geldbeutel schon diese Entscheidung abgerungen hat, dann taucht brennend die eingangs gestellte Frage auf: Mit welcher Sprache beginnen? BASIC ist zwar Standard, die Microsoft Derivate (incl. ST-BASIC) leiden aber an chronischem Mangel moderner Programmierelemente wie Rekursion, lokalen Variablen, Prozeduren und Funktionen. Rühmliche Ausnahmen bilden da GFA- und OMIKRON. BASIC. Fragt man die Herren von der pädagogischen Zunft, so ist meist PASCAL die Antwort auf die Frage nach der Wahl. In der Tat besitzt PASCAL alles was eine moderne (imperative) Programmiersprache braucht und was für das weitergehende Verständnis anderer Sprachen benötigt wird. Leider aber ist der Einstieg für Anfänger so ungeheuer schwer. Tastenverbieger der anfangenden Art haben ja im allgemeinen noch Schwierigkeiten mit der Bedienung der vielen Knöpfe. Und dann sollen auch noch möglichst gleichzeitig die verschiedenen Komponenten eines PASCAL Systems bedient werden als da sind Editor, Compiler, Linker. Kein Wunder, daß sich PASCAL als Anfängersprache immer wieder das Entsetzen der Anfänger zuzieht, bis irgendwann der Knoten platzt und aus Haß Liebe wird. Gewiß, mit TURBO-PASCAL (das in den Schulen stark verbreitet ist) ist der PASCAL Schock stark gemindert worden, aber mit Sehnsucht denkt jeder Anfänger an sein BASIC zu Hause. In dieses Szenario tritt nun mit Fanfaren ALICE, der erste (und meines Wissens einzige) PASCAL Interpreter. Man glaubt seinen Augen nicht zu trauen: PASCAL so einfach zu bedienen wie BASIC?

JA

ALICE kommt auf 2 einseitig formatierten Disketten (nicht kopiergeschützt) mit 2 Handbüchern. Das erste Handbuch ist ein Tutorial zur Bedienung von ALICE. Das zweite Handbuch ist das detaillierte Nachschlagewerk für den Programmierer. Um einen Eindruck von ALICE zu bekommen, kann man getrost auf das Lesen der (englischsprachigen) Handbücher verzichten. Man legt die erste Diskette ein und startet ALICE.PRG. Auf dem Bildschirm erscheint Abb.1. Ja, Sie haben richtig gesehen. Bereits mit dem Start des Interpreters wird die Schablone eines PASCAL-Programms vorgegeben. An diesem Bild können Sie das Arbeitsprinzip von ALICE bereits erkennen, das im folgenden konsequent eingehalten wird. Der ALICE-Editor ist ein an der PASCAL Syntax orientiertes Programm, das keinen Text editiert, sondern die Substitution von Schablonen in PASCAL Termen ermöglicht. Klingt hochgestochen, ist aber ganz einfach. Gehen wir von Abb.1 aus und lassen Sie uns die Entstehung des einfachen Programms verfolgen. Als erstes wollen wir dem Programm einen Namen geben. Dazu gehen wir mit dem Cursor auf die Schablone für den Programmnamen. Man erkennt Schablonen am Unterstreichungsmodus in Abb.1. Es ist egal, wo der Cursor steht, solange er nur irgendwo auf der Schablone steht.

Sobald Sie jetzt beginnen den Namen zu tippen, verschwindet die Schablone und wird durch ihre Eingabe ersetzt. Noch ist die Eingabe nicht akzeptiert, was man an der gestrichelten Unterstreichung des eingegebenen Textes erkennt. Ist der Text eingegeben, überprüft ALICE die Eingabe **sofort** auf

Abb. 1: Die große Schwester Alice

Abb. 2: Überwacht alle Eingaben

Abb. 3: und sagt wo es lang geht

korrekte Syntax **und** Semantik. Wer etwa versucht, in einer Variablendeklaration eine Konstante zu vereinbaren, wird darauf hingewiesen, das ALICE an dieser Stelle des Textes eben eine Variablendeklaration erwartet und nicht die Definition einer Konstanten. Eingeleitet wird diese Überprüfung, in dem man den Cursor von der Schablone entfernt. Im Normalfall verschwindet nun die Unterstreichung, woran man erkennt, daß ALICE keinen Fehler entdeckt hat. Abb.2 zeigt diesen Vorgang im Detail. Das Entfernen des Cursors von der Schablone erfolgt entweder mit den Cursortasten oder mit der RETURN-Taste oder mit der TAB-Taste. Eine Betätigung der Cursortasten bewirkt nichts, außer der erwähnten Überprüfung durch ALICE. Betätigung der RETURN-Taste erzeugt zusätzlich eine neue Schablone der an dieser Stelle zulässigen semantischen Struktur. Auf diese Art wird man sein Programm aufbauen. Mit der TAB-Taste springt man von Schablone zu Schablone vorwärts, überspringt also die PASCAL Schlüsselworte. Fahren wir fort in der Erstellung unseres Programms. Wir wollen einer Variablen i einen Wert zuweisen und diesen ausdrucken. Als nächstes steht also die Deklaration einer Variablen auf dem Plan. Also Cursor auf die **Declarations**-Schablone stellen, var eintippen, Leertaste betätigen und ... die komplette Schablone einer Variablendeklaration steht ausfüllbereit im Editor. Abb.3 zeigt diesen Vorgang wiederum im Detail. Egal welche Anweisung sie eintippen, stets unterliegen Sie der liebevollen Überwachung der großen Schwester ALICE, die sie freundlich aber unnachgiebig auf ihre Fehler hinweist. Es ist einem Anfänger damit wohl unmöglich gemacht worden, die üblichen Fehler zu begehn, wie vergessene Semicolons (ersetzt ALICE selbständig für Sie), Gleichheitszeichen statt Doppelpunkt in einer Variablendeklaration und dergleichen Nadelstiche alter PASCAL Implementationen mehr. Und sollten Sie eine Variable, Funktion oder Prozedur benutzen, die sie bislang noch nicht deklariert haben, so markiert ALICE diese treu und brav, bis sie endlich ihrer vernachlässigten Pflicht nachgekommen sind. Allerdings hat diese strenge Orientierung an der PASCAL Syntax auch ihre Nachteile. Stellen Sie sich vor, Sie haben in einer IF-Anweisung die Bedingung stehen i<j und wollen diese

ändern in i< =j. Dazu gehen Sie mit dem Cursor auf die i<j Bedingung, markieren diese und löschen Sie **vollständig**, worauf die **Condition**-Schablone wieder auftaucht. Dann ersetzen Sie die Condition-Schablone wieder durch die i< =j Bedingung. Das kann im Extremfall zu einer erheblichen Tipparbeit ausarten. Für solche Fälle stellt ALICE zwar die Möglichkeit zur Verfügung, eine Schablone wie Text zu bearbeiten, es bleibt in solchen Fällen jedoch eine etwas unkomfortablere Bedienung als mit einem reinen Texteditor.

Häufig benutzte Strings lassen sich allerdings zum Ausgleich auch auf beliebige Tasten legen und einfach als Macros wieder abrufen. Es ist auch möglich, ein Programm in ein bestehendes zu **mergen**. Die Lesbarkeit kann aber darunter leiden, weil dann plötzlich an zwei verschiedenen Stellen Konstanten-, Typen- oder Variablendeklarationen stehen. Besser ist es, von der Möglichkeit Gebrauch zu machen, mehrere Workspaces gleichzeitig zu eröffnen und mit verschiedenen Programmen zu belegen. Sie können dann von einem Programm zum anderen wechseln, indem Sie einfach das entsprechende Fenster anklicken. Auch Kopieren zwischen den Workspaces ist erlaubt. Mit RUN bringen Sie das im aktuellen Workspace befindliche Programm zur Ausführung, die anderen Workspaces ruhen dann.

Debugging

Nachdem nun Ihr Programm mit Sicherheit syntaktisch einwandfrei im Editor steht, besteht leider immer noch die Möglichkeit, Unsinn programmiert zu haben, d.h. einen fehlerhaften Algorithmus verwendet zu haben. Solchen Fehlern kommt ALICE mit Hilfe seiner vielseitigen DEBUG Möglichkeiten auf die Spur. Es handelt sich bei ALICE wie gesagt um einen **Interpreter** für PASCAL. Sie können also einzelne Anweisungen markieren und **einzeln** ausführen lassen. Natürlich funktioneiert das auch mit Programmblöcken. Sie können das Programm im Single Step Modus durchlaufen und die jeweils berechneten Werte mitanzeigen. Und man kann den Cursor im Editorfenster mitlaufen lassen.

Tabelle 1:
Ausführungszeiten der Benchmarkprogramme in sec.

	ST-BASIC	ALICE	PascalPlus	MEGAMAX C
Fibonacci	–	264	2.8	1.12
Float	191	288	51	52
Erato	100	120	0.8	0.6
Quick	–	140	4	4
Savage	53	238	146	288

Der Sprachumfang

In den USA wirbt Looking Glass Software mit der Kompatibilität zu TURBO-PASCAL. In der Tat ist es mir gelungen, eine Reihe von TURBO-PASCAL Programmen auf dem ST unter ALICE zum Laufen zu bringen. Allerdings ist das nicht ganz so problemlos möglich, wie es die Werbung erscheinen läßt. Auf der zweiten Diskette ist ein Programm Namens APIN.PRG, welches die Konversion vornimmt. ALICE-PASCAL Programme werden nämlich nicht als ASCII-Text abgespeichert, sondern in einer komprimierten Form. Aus diesem Grunde erhalten ALICE-Programme die Extension ★.AP. Die ersten Versuche, Turbo Programme zu konvertieren, scheiterten. APIN druckte eine Fehlermeldung aus und kehrte zum Desktop zurück, ohne dem Benutzer die Chance zu geben, die Fehlermeldung zu lesen. Nach mehreren Versuchen stellte sich dann heraus, daß APIN den Ersatz (. .) für [ ], bzw. (★ ★) für { } nicht mochte. Nach der entsprechenden Änderung mit Hilfe eines Texteditors klappte die ★.PAS -> ★.AP Umwandlung dann problemlos. Auch Programme mit IBM-PC Grafik wurden problemlos übersetzt, bedurften allerdings noch geringfügiger Nachbehandlung wegen der Fensterbehandlung und unterschiedlicher Grafikauflösung. Mit etwas Übung sollte einem damit die Konversion von Turbo-Pascal Programmen auf dem ST unter GEM keine Probleme bereiten. Im OPTIONS-Menü läßt sich ALICE im übrigen auf TURBO-Pascal Syntax oder Standard-Pascal Syntax voreinstellen. Dennoch bleiben ein Paar Inkompatibilitäten, vornehmlich im I/O-Bereich, die sich jedoch an Hand der ausführlichen Dokumentation im Handbuch problemlos beseitigen lassen. Ich erinnere mich noch mit Schaudern an das UCSD-Pascal Konzept, den Benutzer von allem abzuschotten, was im entferntesten mit hardwarenaher Programmierung zu tun hatte. ALICE Pascal erlaubt den Zugriff auf alle GEMDOS, BIOS und XBIOS Routinen. Ja sogar der Interpreter selbst legt seine Arbeitsdaten bei Bedarf bereitwillig auf den Tisch. Damit haben

Abb. 4: Die Zeiten im Vergleich

Tüftler alle Trümpfe in der Hand, Geniales zu zaubern. Als kleines Beispiel finden Sie in allen Benchmark Programmen den Zugriff auf die GEMDOS-Routinen Tsettime() und Tgettime() in den Prozeduren settime und gettime, die für die Zeitmessung herangezogen wurden. Ansonsten unterstützt ALICE den ISO-Pascal Standard.

```
program fibonacci(input, output);
{$I zeit.pas}
{Berechnung der Fibonacci Zahlen}
const
        ntimes = 1;
        number = 20;

var
        i, wert, hr, min, sec : integer;

function fib(x: integer) : integer;
        {Rekursive Berechnung der Fibonacci Zahl}
        {+ Declarations}
    begin
        if x > 2 then begin
                fib := fib(x - 1) + fib(x - 2);
            end
         else begin
                fib := 1;
            end;
    end;
begin
ClrScr ;
writeln('Fibonacci Zahlen');
writeln ;
writeln(ntimes:3, ' Iterationen.');
settime(0, 0, 0);
for i := 1 to ntimes do begin
        wert := fib(number);
    end;
gettime(hr, min, sec);
writeln(hr:2, ':', min:2, ':', sec:2);
writeln('Fibonacci(', number:2, ') = ', wert:10);
end.
```

Listing 1: Berechnung der Fibonacci Zahlen

Die Leistungsfähigkeit

Obwohl ALICE eindeutig auf Anfänger ausgerichtet ist, ist es interessant, etwas über die Leistungsfähigkeit des Interpreters zu erfahren. Schließlich möchte man auch als Anfänger nicht unbedingt im Kopf schneller rechnen als der Computer. Direkter Konkurrent für ALICE ist natürlich das ST-BASIC, welches mit dem Computer ausgeliefert wird. Könner schätzen auf dem ST ja im allgemeinen C. Deshalb ist es vielleicht interessant zu sehen, wie ALICE im Vergleich zur „Muttersprache" des ST abschneidet. Ich habe den MEGAMAX Compiler zum Vergleich herangezogen. Die Benchmark Programme wurden [1] entnommen und für ALICE angepaßt. Listing 1–5 zeigen die ALICE-Programmlistings. Es handelt sich im einzelnen um folgende Programme:

FIBONACCI

Dieses Programm berechnet eine Zahl aus der Fibonacci Folge. Diese ist rekursiv definiert nach der Regel $a_n = a_{n-1} + a_{n-2}$, wenn $n > 2$ ist, $a_1 = a_2 = 1$. Damit wird also das Rekursionsverhalten des Interpreters/Compilers getestet. Dieses Programm läuft natürlich nicht in ST-BASIC.

FLOAT

Mit diesem Programm werden die Gleitkommaroutinen für Multiplikation und Division getestet.

ERATO

Das Sieb des Eratosthenes. Findet Primzahlen nach der o.g. Methode. Testet die Schleifen- und Feldroutinen. Benutzt nur Integerarithmetik.

QUICK

Rekursives Quicksort Programm. Sortiert ein zufällig erzeugtes ganzzahliges Feld der Größe nach. Testet Rekursionsverhalten und Entscheidungen. Läuft wegen der Rekursion nicht in ST-BASIC.

SAVAGE

Dies ist der gemeinste Test. Er liefert Aussagen über die Schnelligkeit und Genauigkeit der transzendenten Funktionen und der Wurzelfunktion. Da in der Zuweisung immer eine Funktion und ihre Umkehrfunktion gepaart vorkommen, muß der Wert am Schluß genau so groß sein wie die Zählvariable. Die Abweichung sagt etwas über die Genauigkeit der entsprechenden Routinen aus.

Die Ausführungszeiten sind in Tabelle 1 zusammengefaßt. Abb.4 zeigt noch einmal die Ergebnisse graphisch dargestellt. Es zeigt sich, daß ALICE von der Geschwindigkeit her dicht bei den entsprechenden BASIC Programmen liegt. Es sind halt beides Interpreter. Die Ausführungszeiten entsprechen im allgemeinen den Erwartungen. Natürlich fegen die Compiler ab wie die Feuerwehr. Lediglich das Ergebnis für den Savage-Test überrascht. ST-BASIC am schnellsten? Liegt vielleicht an der Genauigkeit? Tabelle 2 macht das Erstaunen perfekt: Der Geschwindigkeitsunterschied zwischen BASIC und ALICE läßt sich zwar erwartungsgemäß auf den Genauigkeitsunterschied zurückführen, bleibt aber die bittere Erkenntnis, daß der MEGAMAX Compiler im Bereich der transzendenten Funktion offensichtlich Schwächen aufweist.

Tabelle 2: Abweichung des berechneten Wertes vom erwarteten Wert 5000 im Programm SAVAGE.

ST-BASIC:	-3547.94
ALICE:	-9.304E-7
PascalPlus:	-9.147
MEGAMAX C:	-2276.545

```
program float(input, output);
{$I zeit.pas}
{Test der Real-Arithmetik}
const
        const1 = 3.141597E0;
        const2 = 1.7839032E4;
        count = 10000;

var
        i, hr, min, sec : integer;
        a, b, c : real;

begin
ClrScr ;
writeln('Floating Test');
writeln ;
settime(0, 0, 0);
a := const1;
b := const2;
for i := 1 to count do begin
        c := a*b;
        c := c/a;
        c := a*b;
        c := c/a;
        c := a*b;
        c := c/a;
        c := a*b;
        c := c/a;
        c := a*b;
        c := c/a;
        c := a*b;
        c := c/a;
        c := a*b;
        c := c/a;
    end;
gettime(hr, min, sec);
writeln(hr:2, ':', min:2, ':', sec:2);
end.
```

Listing 2: Float Test

Da MEGAMAX die Möglichkeit bietet, unter hinzulinken der Datei DOUBLE.L die Routinen mit doppelter Genauigkeit zu benutzen, habe ich natürlich von dieser Möglichkeit Gebrauch gemacht. In der Tat hat MEGAMAX dann eine vergleichbare Genauigkeit (−7E−6), aber die Ausführungszeit ist auf 990 sec angewachsen! Offensichtlich besitzt ALICE ganz ordentliche Routinen für Gleitkommaarithmetik.

Stand-alone Anwendungen

So vorteilhaft sich Interpreter bei der Bedienung auch darstellen, sowohl mit der Geschwindigkeit als auch mit der Weitergabe eigener Programme, gibt es Schwierigkeiten. Wie wir oben gesehen haben, sticht das Geschwindigkeitsproblem bei ALICE nicht so sehr. Wohl aber die Möglichkeit, eigene Programme weiterzugeben. ALICE ermöglicht zwar, Programme sowohl im komprimierten ★.AP Format als auch als Textfiles abzuspeichern, aber damit kann der Benutzer lediglich den Source Code seines Programms weitergeben. Dies hat den Nachteil, daß der Empfänger des Programms im Besitz eines Interpreter oder Compilers der gleichen Sprache sein muß, wenn er das Programm benutzen will. Weiterhin gibt der Autor mit dem Programm seinen mühsam erarbeiteten Quellcode frei, was natürlich zum „Lernen" anregt. Das erste Problem ist durch sogenannte Run-Only Interpreter gelöst. Im Gegensatz zu GFA, die für ihren BASIC-Interpreter gleich eine Run-Only-Interpreter Version zum Vertrieb freigeben, konnte sich Looking Glass Software zu dieser Freizügigkeit (noch) nicht entschließen. Ernsthaft Interessierte sollten deshalb den Run-Only-Interpreter APRUN.PRG gleich mitbestellen, um Freunden oder der PD den Genuß selbstgeschriebener AP-Programme zu ermöglichen. Offensichtlich hat man aber von vornherein eine andere Strategie ins Auge gefaßt. Da auch die Autoren von ALICE um die Geschwindigkeit von Interpretern wissen, haben sie im Filemenü den Menüpunkt **Invoke Compiler** mit aufgenommen, der automatisch das aktive Programm als Text abspeichert, den PASCAL Compiler aufruft und anschließend zu ALICE

```
program SieveOfErastothenes(input, output);
{$I zeit.pas}
{ Calculates prime numbers up to 2 * size + 3 }
const
        size = 1000;

var
        hr, min, sec : integer;
        i, prime, k, count, iter : integer;
        flags : array [0..size] of Boolean;

begin
settime(0, 0, 0);
count := 0;
{ Initialize the array of flags }
{ Start off with all numbers being prime }
for i := 0 to size do begin
        flags[i] := true;
    end;
for i := 0 to size do begin
        { Cross non-primes off the list }
        { Found a number that hasn't been crossed }
```

(Fortsetzung auf S. 136)

(Fortsetzung von S. 135)

```
{ off, so it must be prime }
if flags[i] then begin
        prime := i + i + 3;
        { Print out the prime }
        {
                writeln(prime);
        }
        k := i + prime;
        { Cross out all the multiples of this }
        { prime }
        while k < size do begin
                flags[k] := false;
                k := k + prime;
            end;
        count := count + 1;
    end;
end;
gettime(hr, min, sec);
writeln(hr:2, ':', min:2, ':', sec:2);
writeln(count, " primes");
{ Write the final number of primes }
end.
```

Listing 3: Das berüchtigte Sieb des Erastothenes

```
program quicksort(input, output);
{$I zeit.pas}
{Sortiert ein Array mit real-Zufallszahlen}
const
        maxnum = 100;
        count = 10;
type
        feld = array [1..maxnum] of integer;
var
        i, j, temp, hr, min, sec : integer;
        buffer : feld;

procedure quick(lo, hi: integer; var base: feld);
        {Rekursiver Quicksort}
        var
                i, j, k, pivot, temp : integer;

    begin
        if lo < hi then begin
                j := hi - 1;
                pivot := base[hi];
                i := lo;
                while i < j do begin
                    while (i < hi) and (base[i] <= pivot) do begin
                            i := i + 1;
                        end;
                    while (j > lo) and (base[j] >= pivot) do begin
                            j := j - 1;
                        end;
                    if i < j then begin
                            temp := base[i];
                            base[i] := base[j];
                            base[j] := temp;
                        end;
                    end;
                if base[i] >= base[hi] then begin
                    temp := base[i];
                    base[i] := base[hi];
```

(Fortsetzung auf S. 137)

zurückkehrt. Im Handbuch und im AP.INI File sind die Daten für den OSS-Compiler aufgeführt. Trotzdem dürften hier Probleme auftauchen, weil der Zugriff auf das GEM bei ALICE eben doch etwas anders abläuft als bei den PASCAL Compilern für den ST. Diese Option ist wohl eher für den IBM-PC und Turbo-Pascal gedacht gewesen.

Zusammenfassung

ALICE Pascal ist ein leistungsfähiger PASCAL Interpreter. Turbo Pascal Programme sind mit minimalen Änderungen lauffähig. Der an der PASCAL Syntax orientierte Editor verhindert weitestgehend Syntaxfehler bereits beim Eintippen des Programmtextes und stellt damit die ideale Lernhilfe für PASCAL-Anfänger dar. Alle Fehlermeldungen werden reichlich (um nicht zu sagen schwatzhaft) kommentiert (leider in Englisch). Zugriff auf GEM ist möglich und z. Teil für Anfänger erheblich vereinfacht. Wer interessiert ist, PASCAL zu lernen, sollte ALICE in Betracht ziehen. Als Könner sollte man allerdings die Kosten eines guten PASCAL Compilers gleich mit einkalkulieren. Oder man erweitert seinen Horizont und betrachtet PASCAL als Sprungbrett für C und kauft sich einen C-Compiler.

Dr. Sarnow

Produkt

ALICE. The Personal PASCAL.

Hersteller

Looking Glass Software Limited
4 King St. N.
terloo, ON
N2J 2X8
Canada

Literatur

[1] BYTE, July 1987, Vol.12, No. 8, p. 101 ff.

Preis: 79,95 $ (US) + 15 $ Fracht

(Fortsetzung von S. 136)

```
                        base[hi] := temp;
                        quick(lo, i - 1, base);
                        quick(i + 1, hi, base);
                    end;
                end;
            end;
        begin
        ClrScr ;
        writeln('Quicksort');
        writeln ;
        settime(0, 0, 0);
        for i := 1 to count do begin
            for j := 1 to maxnum do begin
                buffer[j] := round(10000*random );

        end;
    writeln('Buffer voll, Iteration ', i);
    quick(1, maxnum, buffer);
  end;
gettime(hr, min, sec);
writeln(hr:2, ':', min:2, ':', sec:2);
end.
```

Listing 4: Schnell Sortiert – Der Quicksort

```
program savage(input, output);
{$I zeit.pas}
{Test der arithmetischen Genauigkeit}
const
        iloop = 5000;

var
        i, hr, min, sec : integer;
        wert : real;

function tan(x: real) : real;
    {Berechnet den Tangens eines reellen Wertes}
    begin
        tan := sin(x)/cos(x);
    end;
begin
ClrScr ;
writeln('Savage Benchmark');
settime(0, 0, 0);
wert := 1.0;
for i := 1 to iloop do begin
        wert := tan(arctan(exp(ln(sqrt(wert*wert))))) + 1.0;
    end;
gettime(hr, min, sec);
writeln(hr:2, ':', min:2, ':', sec:2);
writeln('Abweichung=', wert - iloop - 1);
end.
```

Listing 5: Savage: der Genauigkeitstest – hier fallen die Hüllen

```
const
        gemdos = 62;
        tgettime = 44;
        tsettime = 45;

procedure settime(hr, min, sec: integer);

        {Setzt die Uhr}
        var
                zeit : integer;
    begin
        zeit := sec and 63;
        zeit := zeit + ((min shl 6) and 4032);
        zeit := zeit + ((hr shl 12) and $1F000);
        CProc(gemdos, tsettime, zeit);
    end;
procedure gettime(var hr, min, sec: integer);
        {Gibt die Systemzeit zurueck}
        var
                zeit : integer;

    begin
        zeit := 2*CIntFunc(gemdos, tgettime);
        sec := zeit and 63;
        min := (zeit and 4032) shr 6;
        hr := (zeit and $1F000) shr 12;
    end;
```

Listing 6

BIETE HARDWARE

Typenraddrucker Brother HR-15 XL mit autom. Einzelblatteinzug zu verkaufen, Preis VHS. Tel. 0441/13606 o. 84979

SF 354 neuw. 150 DM, Org. Atari Netzteil 40 DM, CSF-Gehäuse DM 80 T. 089/8347916

★Einzelstation 349 DM★
★Doppelstation 659 DM★
★Netzt. eingeb., NEC 1036A★
★★ kompl. Anschlußfertig ★★
★ -> SF354 Umbau 288 DM ★
FISCHER-Computer 06151/51395

520ST+, Monitor SM124, 360KB v. 720KB Laufwerk, Romtos, Softw. VB 2000. T. ab 18 Uhr 08862/7151

★★★★Superangebot!!★★★★
Eprommer für ATARI ST DM 249 (Fertiggerät incl. GEM-Software)
ABC-SYSTEMS ★ Schillerstr. 56 4432 Gronau ★ Info anfordern!

Diskettenzugriffe bis zu 100% beschleunigen! Durch kinderleichten Einbau unseres Speedchips! Kein Löten! Ausführliche Anleitung! Gratisinfo: 0421/6589278 Marco Meyer / Gerhard-Rohlfs-Straße 54c / 2820 Bremen 70

Thomson-Farbmonitor + Aladin zu verkaufen: (02636) 2137.

Floppystation für ATARI-ST 2x1 MB mit eingeb. Netzteil nur DM 700. Tel. 08142/2727

260ST SF354 Maus 10 PD-Disk ev. SM124, Garantie 08765/628

>> BLITTER-TOS <<
Gratisinfo: 0421/6589278 Marco Meyer / Gerhard-Rohlfs-Straße 54c 2820 Bremen 70

★ **ATARI ST Diskettenstation** ★
1x3,5" 726KB Abm. 105x30x220mm zusätzlich eing. Floppybuchse (NEC) eing. Netzteil **nur 359,–**
2x3,5" 1,4 MB **nur 649,–**
NEC FD 1036A für ST nur 238,–
Speichererweiterung Lötfrei u. Steckbar 229,– FASTROM 19,90
★ Weiteres Zubehör günstig ★
★ DELO Computertechnik ★
★ Tel: 0231/339731 o. 331148 ★

Matrix-Drucker DMP2000+Kabel für Atari ST DM 400 – 0931/83509

520ST+ TOS, im Eprom+Maus+ Monitor SM124+Floppy SF314 (DS)+Basic u Logo u... org Handb. Tel. 02171/89424 (Opladen) ab 19°° 1700 DM für Selbstabholer

ATARI-Echtzeituhr akkugepuffert Einbau ohne Löten DM 98 Romport nicht belegt T. 02337/1239

Blittertox auf Eprom brennt T. 05171/3723

ST+ SF314, SM124, **STAR NL10**, ca. 70 Disks, Bücher, Maus 02204-62776

520ST+, 1MB, ROM-TOS, SF314, SF354, SM124, PC-Gehäuse, Uhrmodul, Maus, Software, Bücher, 2100 DM VB. DIN A3 Citizen MSP15 Drucker, noch 10 Monate Garantie, sehr guter Zustand, VB 1000 DM ab 18 Uhr 07152/45574

DRUCKER STAR SG-15 DIN A3 DM 900,– Tel. 089/3108872

Atari-Farbmonitor SC1224 620 DM SF 354 110 DM; Tel. 0212/549595

520ST+, NEC 2x 720 KB Doppellaufw. zus. 1098,- Tel. 02261/56786

Die Atari-Mailbox: 0571/710141

SF354 - 120 DM★PC-Gehäuse von CSF VB 90,– 0221/638619 ab 17h

Atari 1040 + SF314 + PC-Geh. + Monitor viel Softw. dBaseII neuw. DM 4500,– jetzt VB 1900,– Tel. 06021-57717

BIETE SOFTWARE

P6 Fonteditor/download, 2gr/math. Fonts (NLQ) + 1Word-Tr., Kaiser Erdingerstr. 24, 8050 Freising

! Bundesliga auf Atari ST, Mausbedienung, Analyse m. Disk 29,– Thielmann, Dieter 02903/2551 !

Totosystem m. Gewinngarantie Voll GEM-gesteuert, Info gegen Freiumschlag. Orig. GST-C-Comp. wg. Systemw. 089/8888361 18.00

Neuheit! Codec V2.0 (GEM-Anw.) Verschlüsseln Sie vertrauliche Dateien/Prg. byteweise mit sicherstem Verf. Für Unbefugte n. reidentifizierbar. DM 39,– NN. Chiffre F8710

DBASE II Original mit Anleitung gegen Gebot Tel. 0202/86886

Protext DM 75,– Tel. 07158-6058

MCC Pascal 70,– 0721/519780

Super-Lernsystem mit integrierter Karteiverwaltung! Voll unter GEM! Internationale Zeichen! Viele Lernmodi! Top Handbuch! Gratisinfo: 0421/6589278 Marco Meyer / Gerhard-Rohlfs-Str. 54c/282 Bremen 70

PUBLIC DOMAIN Kopierservice über 200 PD-Disk ★ Gratiskatalog Interessante Preise ★ einseitig 5,50 DM ★ zweiseitig 6,50 DM M. Schönfelder ★ Tel.: 02954-1050

■■■ Individuelle ST-Software ■
■■ incl. Einführungs+Pflege! ■■
■Holger Zwar, ☎02246-4910■■■

Wärmebedarfsberechnung DIN4701
KZahlberechnung DIN4108
Wärmeschutznachweis WäSchV
Dipl. Ing. V. Koch, Am Mehnacker 11 3563 Dautphetal 3, Tel. 064687652

$ mit Aktien Gewinne machen $
$ Depotverwaltung DM 178 $
der Weg zur erfolgreichen
Analyse von Wertpapierkursen
A. Nemetz v. Bergmannstr. 39
6200 Wiesbaden

C-Auftrag / C-Text + Version 2.11 DM 350,– Tel. 07371/7267

Verkaufe Original GfA-Basic Interpreter + Compiler + Buch zus. 300,– DM Tel. 06858/8602

■■ Public-Domain DM 5,00 ■■
■ PD-Angebot dieser Ausgabe ■
■ Einzeldiskette SS DM 5,00 ■
■ Pakete 1–10, 11–20, usw. ■
■ auf je 10 Disk. DM 45,00 ■
■ Porto u. Verpack. DM 3,00 ■
■ V-Scheck o. NN (+DM 3,20) ■
■ E. Twardoch, Kais.-Wilhelm- ■
■ Straße 88 – 1000 Berlin 46 ■

★★ PD-Service DM 6,00 ★★
Angebot wie PD dieser Ausgabe Einzeldiskette SS DM 6,00
Pakete 1–10, 11–20, 21–30, 31–40 und so w. je DM 38,00 auf je 5 Disketten DS
Preis inkl. Disk zzgl. Porto u. Verp. DM 5,00 (Ausl. DM 10,00) NN plus 1,70 besser V-Scheck
I K S, Schönblickstr. 7
7516 Karlsbad-Mu
ab 18 Uhr 07202/6793

PD-Software ab 2,– DM/Disk Tel: 02721/2432 ab 16 Uhr

Berlin: Public Domain Software, 140 Disks je DM 7,– sofort zum Mitnehmen, Versand in die BRD. Gratisinfo bei: Behrendt's Computershop Fürbringerstr. 26, 1000 Berlin 61, Tel. 691 76 66

■■ Tore–Punkte–Meisterschaft GEM-Bundesliga-Prg, 4-28 Clubs Deutsche-, Engl.-, Heim-, Ausw.-, Halbz.-Tabellen, DFB-Spielpl., 29,–, Info 5,– Tobias Becker Dorstfelder Hellweg 92, 46 Do 1

LOTTO-ST alle Ziehungen bis Sommer '87, Tipvorschlag, Häufigkeit, Wiederholung, Kombinationen (3'er etc), Eingabe, Ausgabe, 49,– DM Tel: 09343/4462

Original TDI-Modula2 + Toolkit 3.00a, sFr 300 statt DM 448! CH 054-651691

>>SUPER-PD-SOFTWARE<<
Gratisinfo: 0421/6589278 Marco Meyer / Gerhard-Rohlfs-Straße 54c 2820 Bremen 70

Neu! 1987+88 Lohnsteuerjahresausgleich+Est-Berechnung+Lst+ Est-Tabellen / 30 DM Vork. Höfer Grunewald 2a, 5272 Wipperfürth Tel. 02192/3368 Backup!!!

★★ Public Domain DM 5,00 ★★
★ über 160 Disketten (voll) ★
★ auf Markendisketten ★
★Buchhandlung Werner Finke★
★Kipdorf 32, Tel. 0202/454220★
★D-5600 Wuppertal 1(Elberfeld)★

Orig. Programme mit Handbuch z. halben Preis: Beckertext 99 DM, dBase II 170 DM, Heim-Manager 49 DM, Ahnen-Verwaltung 30 DM. Beer, Parkstr. 8, 5860 Iserlohn, Tel. 02371-61787, abends

Druckertreiber f. Wordplus/NL-10 wesentlich erweitert, vollständige Ausnutzung der NL-10 Mögl. z. B.: sub/superscrips in Normal-NLQ ü. 40 neue Befehle, 7 Grafikmodi m. ausf. Anl. (~40kB), **HEX+CFG** f. IBM/Centr/Eps 30 DM incl. Disk Zahl. n. Erhalt. K. Plüher K.-F.-Friesen Str. 26, 4690 Herne 1

★ **ST-Statistik unter GEM mit** ★ Update-Service: deskrip. Daten para+nonparametr. Tests, Korr.-tabs, Varianz-, Faktor-, Cluster Analysen ! Übernahme in Textprg DM 249,– / DemoDisk DM 30,– Scheck/NN Dipl. Psych. M. Prall Isestr. 57, 2000 Hamburg 13

ST-MAIL – DAS Mailboxprogr. für Atari ST!! Info bei Software-Entwicklung und -Vertrieb M. Nitschke, Bahnhofstr. 21, 6330 Wetzlar

„Die Videothek" (sw) Verwaltung Ihrer privaten Filme mit Drucken von Etiketten, Listen u.v.m. Vorkasse incl. Disk. DM 49,– Dietmar Rathke, Tintengasse 1 7253 Renningen

PD-Grafik-Bilder für ST von NEO, Degas u. a. Katalog V. Frey, Rheinstr. 12A, 6538 Münster-Sarmsheim

Statik- u. Mathem.-Progr., sehr schnell. Dipl. Ing. Jürgen Bullmann Zur Hindenburgschleuse 3, 3000 Hannover 71

■■ STOP! Hier geht's rund! ■■
Public-Domain Software: Atari ST und im ALADIN-Format. Gratis-Info bei: ■ Carsten & Marcus ■ Postfach 650602, 2000 Hamburg 65

Im Tiefflug durch die PD-Welt!
★★★ Tel.: 09343/8269 ★★★

Wärmebed. DIN4701 + K-Zahl DM 110; Rohrnetzber. DM 60; Demodisk DM 10; nur Vorkasse von J. Binder, Eichendorffstr. 15 5030 Hürth

■ >>SENSATIONELL<< ■
■ Über 280 PD-Disks, enorm ■
■ preisgünstig, Blitzversand ■
■ uvm. Gratisinfo bei RALF ■
■ MARKERT, Balbachtalstr. 71 ■
■ 6970 Lauda · Tel. 09343/8269 ■

★★PD-GRATIS-AKTION!★★
Die schönsten PD-Programme können Sie bei uns schon gratis bekommen! Wie & Wo? R. MARKERT Balbachtalstraße 71 · 6970 Lauda

Schweiz: Publ. Domain Software PD01-PD51 aus ST-Comp-Zeitschr. Zwei Disc auf einer Markendisc z.B: (1+2) (3+4) (5+6) (15+16) etc. sFr. 10 pro Disc, Check od. BAr PD-Soft, Postfach 8, 8602 Wangen

■■ Public-Domain DM 6,00 ■■
■ PD-Angebot dieser Ausgabe ■
■auf Mark.-Disk SS je DM 6,00■
■ Verp. DM 5,00, ab 20 St frei■
■V.-Scheck o. NN (+DM 1,70)■
■ J. Rangnow, Talstr. 8 ■
■ 7519 Eppingen (HN) ■
■ 07262/5131 ab 18 Uhr ■
■■■ PD-Express - Versand ■■■

Wärmebedarfs./KZahlberechnung DIN4701/83 GEM Bedienerführung Dipl. Ing. V. Koch, Am Mehnacker 11 3563 Dautphetal 3, Tel. 064687652

FIBU, Buchführung für Freiberufler, Vereine, Privat, komf. Buchen, Saldenlisten, G+V, Bilanz, Statistik, MWSt, alles mögliche. Nur 59,– DM!!! Info: O. Kuschek, Mülheimer Mühle, 5378 Blankenheim, 02449/1003

MITGLIEDER-Verwaltung für Vereine und Sontige; Serienbrief, Listgenerator, Beitrag, Mahnen, Abbuchen etc. max. 1000 Mitglieder. 69 – DM. Info: O. Kuschek, Mülheimer Mühle, 5378 Blankenheim 02449/1003

★ Die beste PD für Ihren ST ★ EUSAX PD-Service 4018 Langenfeld, Haus Gravener Str. 85

Verkaufe: EasyDraw 250,-, DEGAS 100,-, GFA DRAFT 200,-, TOS-ROM 80,-. Tel. 0221-706012

Public-Domain-Blitzversandt!
Riesenauswahl! Preise:
ab DM 4,– einseitige Disketten
ab DM 6,– doppelseitige
Inclusive Diskette!!!
Auch alle ST-Disks! Auf Doppeldisks beliebig kombinierbar!
Gratisliste anfordern bei:
A. Gauber Software, Buhlstr. 16a
7505 Ettlingen, 07243/31828

■ Datenverwaltung DM 95,– ■
■ Videocass.archiv DM 50,– ■
■ TOP-Superhirn DM 30,– ■
■ Musikcass.archiv DM 45,– ■
■ Kostenl. Info: Peter Wittmann■
■ Daimlerring 28, 6729 Rülzheim■

Von Lehrer für Lehrer! Notenverw.prg. Markbook plus in GEM. Über 20 versch. Funkt. f. alle gäng. Notensyst. Spez. vers. auf Wunsch; Info gegen Freiumschlag.
H J. Merkel/Nahestr. 28/6600 SBR

STatistik★Analyse med/biol. Dat Nichtpar. u. par. Tests, Varianz-Analyse Datenüb. aus Datenbank Tel. 09131/209401 n. 18°° ★ST★

BUSINESS-BÜRO-MASTER
Inc: Serienbrief, Kundendatei, Rechnung, Mahnung, Kalkulation, Grafik, Lager. Info: 1,10 DM bei Arkosoft · Postfach: 4824 · 48 BI 1

PD-Software, direkt vom Autor, die es nur bei mir gibt. Riesen-Angebot, alle Prg. einzeln auswählbar. Der absolute Hit:s nur 1,5 Pf pro KByte!! Endlich wird nach SOFTWAREMENGE und nicht nach DISKPREIS abgerechnet! Gratisinfo: Arne Zingel, Vermehrenring 11a, 2400 Lübeck 1

★ PD-SOFTWARE DM 5,00 ★
Spiele, Musik, Anwendungen
Info: E. Twardoch,
Kaiser-Wilhelm-Str. 88, 1 Berlin 46

PUBLIC 190 Disk's DOMAIN
U zus. alle PD's I
B aus „ST-Computer"! A
L –★– M
I Gratiskatalog: O
PUBLIC –★– DOMAIN
Hans Frey
Bruckenäcker 3
7000 Stuttgart 80

Leerdisketten? Tel.: 0431/569216, zum Wochenende: Tel.: 0481/72839

Vokabel-Programm, Info 2,–; Demo-Disk. 10,–; H. Michels, Föhrenweg 11, 3400 Göttingen

Aktien-Prog. gesucht und biete hochw. selbstgeschr. Programm, Schneider, Ignaz-Jaspers-Straße 12 4830 Gütersloh 1, 05241/55710

Verk. Org. Karate Kid II, Major Motion, Goldrunner, Monostar, je DM 50, Verlängerungskabel für Easyprommer DM 25 T. 08725-1373

Text und Literaturverwaltung! Voll GEM, schnell, da in C. 75 DM. Info: P. Jodda, Heidkamp 18, 3100 Celle.

– Branchensoftware Kfz.-Handel
– Adressystem, 1st Word+ komp.
Unter Tel. 06144-41505

PLOT_ST für Pro-FORTRAN! Einfachste Ausgabe grafischer Darstellungen auf Bildschirm und Drucker. Mehrere Funktionen gleichzeitig, sowie beliebige Punktwolken darstellbar! Komfortabel wie auf einem Großrechner.
DM 95,–. Handbuch DM 15,– (wird verrechnet); Info gratis.
Bestellung/Info bei: M. Gamer Friedrichsring 26, 6050 Offenbach

assoziative Datenbank Themadat schnelle, assoziativ arbeitende Datenbank 49,- DM, mit Textmaskengenerator 74,- DM plus 5,- DM Porto (V-Scheck) H. v. Tryller 3200 Hildesheim, Steinbergstr. 6

SUCHE HARDWARE

SF 314 GES. (RAUM ESSEN) Chiffre G8710

Harddisk/Plotter 07158-6058

SUCHE SOFTWARE

ST Vision-Userclub, PD-Library, regelm. Zeitung mit Top-News ST Vision, Pf. 1651, 6070 Langen

Suche Mailboxprg. 089/3084026

CAD-Programm Tel. 07158-6058

Original-ST-Software: Signum, GFA Publisher, Wordplus, Psion Chessbase, Skyplot: 0 26 36 / 21 37

VT220-Terminal-Emulation für ST. Anr. abends! 069-316452

Suche alles über C, Literatur, Programme T. 05171/3723

Finanzbuchhaltung mit Anleitung Tel. 0212/62489

SUCHE HARDWARE

Kontakt! GFA, PD, keine Raubkopierer! Clubs gesucht!
Habe selbst PD-Soft + Digi-Bld
Phone: 07321/63879 ab 20°° Uhr

Aktien-Prog. gesucht und biete hochw. selbstgeschr. Programm, Schneider, Ignaz-Jaspers-Straße 12 4830 Gütersloh 1, 05241/55710

VERSCHIEDENES

DDD-Mailbox = der PD-Bagger! 0211-719261, 24h, 300/1200,8N1 Fast-TOS 20,–, ROM-TOS 100,–! DDD Benrather Schloßallee 73 4000 Düsseldorf 13 0211-717141

Mega-Team. Der Preisdiscounter z. B. Textverarb. 24,95 DM ect. Infos: Mega-Team Bannizastr. 31 4250 Bottrop Tel. 02041/94842

Soft- u. Hardw. f. Atari + Commodore sehr preisgünstig! Strathmeier Postf. 101215, 4978 B. Oeynhausen

Big Brotherhood of Swap ST&Ami Info auf Clubdisk Nr. 1 gegen 10 Mark vom Astro C. Klub Karlsmark 3, 2262 Leck, Tel. 06662/4557

Digital. Sounds/Sprache für Ihre Progr.!!! Günstig! 025291496

Software-Programmierer für C, GFA-BASCI oder Assembler ges. T. Zimmermann, Steinackerstr. 12 7520 Bruchsal, Tel. 07251/14920

Bevor sie in DTP investieren, sollten Sie unser Angebot kennen! Wir lesen Ihre Disk, erst. Druckvorlagen, drucken m. Laser u. Offset. Tel. 06027/75791

★Super Soft- & Hardwarepreise★
Fonteditor u. Konverter DM 79,-
3D-Video-Studio ..... DM 148,-
3D-Demoanimation (4 Disks) 25,-
PD-Softwae, Disk. ab DM 4,50
10 3.5"-Disketten, 1 DD DM 26,-
STAR NL 10 m. Interf. DM 589,-
Versandliste gratis.
Expert Software · Pestalozzistr. 6
4350 Recklinghausen 02361/36267

! Brennen Eproms für Atari !
!ROM-TOS mit Änderungen aus!
! ST und CT nach Wunsch !
!Programme für EPROM-BANK!
! Tel: 08191/8205 o. 08248/1663 !

KONTAKTE

Atarineuling 1040+SM124 sucht Software- und Erfahrungsaustausch mit GfA- und PASCAL-Freaks R. Hartnagel, Falkenstraße 12 8501 Eckenhaid

TAUSCH

'L'-Porgrammierung!!! Suche Leute zum gemeinsamen profess. Einstieg in C u Ass W. Lampert Hoistener Str. 35, 404 Neuss 476130

Anzeigenschluß
für
Heft 12/87
ist der
20. 10. 1987

Anzeigenschluß
für
Heft 12/87
ist der
20. 10. 1987

Leserecke & Leserbriefe

Leser → Leser

Betr.: Leserbrief von Herrn Pillmann aus der ST 9/87

Sie schreiben, daß Sie Ordner oft zweimal anklicken müssen, bevor „so richtig die Sonne aufgeht". Kurzfristige Abhilfe (z.B. in Basic): Klicken Sie vor dem Start des Basic-Interpreters den betreffenden gefährdeten Ordner an (also aus dem Quellcode-Ordner). Dann müßte es aus dem Interpeter heraus funktionieren.
Die Ursache dieses Mysteriums ist mir nicht bekannt, ich habe aber eine Vermutung. Früher hatte ich bei diversen Disketten ähnliche Probleme – die waren mit FATDISK formatiert. Nun hatte die FATDISK immer die unangenehme Angewohnheit, immer die gleiche Seriennummer auf die Diskette zu schreiben – was u.U. verhindert, daß TOS Diskettenwechsel erkennt. Auch ich besitze ein Programm namens SUPERFORMAT, das den gleichen Fehler macht (was nichts zu bedeuten hat), anscheinend nennt jeder sein Format-Programm gleich SUPER). Seit ich mein eigenes HYPERFORMAT (welch origineller Name...) verwende, das bei jedem Formatieren eine zufällige Seriennummer erzeugt (so wie TOS es auch tut), sind Effekte wie der beschriebene nicht mehr vorgekommen. Vielleicht probieren Sie es einfach mal damit.

Claus Brod, Marktheidenfeld

Gewöhnlich tritt der beschriebene Fehler (Nichtöffnen von Ordnern nach Diskettenwechsel auf, wenn die Diskette mit FATDISK oder SUPERFORMAT formatiert wurde. Diese Programme machen beide einen entscheidenden Fehler: Beim Erzeugen des Bootsektors mit Hilfe der XBIOS-Funktion Protobt (18) wird die Seriennummer $1000000 übergeben. Nun ist in allen Handbüchern nachzulesen, daß man, wenn man eine zufällige Diskettennummer bekommen möchte, eine Zahl größer als $1000000 übergibt! So tragen alle Disketten, die mit solchen Frmatierroutinen beschrieben wurden, die Nummer 000000. Abhilfe schafft:

- Die Diskette vom Desktop aus zu formatieren
- eigene Formatierroutine schreiben
- mit Diskettenmonitor bereits zerstörte Disketten bearbeiten. Zu ändern sind dann die Byte 08–OA (1. Byte = 00).

Sven E. Knop, Aachen

Leser → ST

Betr.: Artikel „Patch für GFA", ST Computer 9/87

In dem Bericht erweckt der Autor Claus Brod den Eindruck, daß unser Programmierer, Herr Ostrowski, unsauber programmieren würde. Weiterhin legt Herr Brod dar, daß die Funktion kbshift(–1) von Herrn Ostrowski hätte benutzt werden sollen.
Richtig ist, daß Herr Ostrowski aufgrund einer Nachfrage bei ATARI Deutschland diese einzige im GFA BASIC enthaltene undokumentierte Adresse benutzen mußte, weil Systemaufrufe im Interrupt nicht funktionieren, also auch nicht die Funktion kbshift(–1). Unsere neueste Version 2.02 des Interpreters sucht selbständig nach der undokumentierten, aber benötigten Betriebssystemvariablen _kbshift und kann daher mit sämtlichen alten und neuen TOS-Versionen betrieben werden.

Rolf Hilcher, GFA Systemtechnik GmbH

Betr.: Artikel „TeX auf dem ATARI ST", ST Computer 7/8/87

Die Erstellung von Textdateien für die Bearbeitung mit TeX (bei mir von TOOLS) mit Beckertext ist für folgender Vorgehensweise völlig unproblematisch:
Voreinstellung: AUTO-TRENN abschalten, Textbreite 80 Zeichen (am besten Zeichensatz schmal wählen), schreiben im BRIEF-MODUS! Die Textdateien werden als ASCII-File mit Extension .TEX abgespeichert. Zusätzlich werden dieselben Dateien als TEXT mit der Extension .TXT abgespeichert.
Bei Änderungen im Text werden nur die .TXT-Dateien als TEXT eingelesen und mit Beckertext nach demselben Verfahren erneut bearbeitet. Das doppelte Abspeichern vermeidet das Einstreuen von zusätzlichen RETURN-Befehlen bei alleiniger Arbeit mit dem ASCII-File; es sind bei der Textbearbeitung alle Vorteile von Beckertext nutzbar (z.B. Funktionstastenbelegung mit TeX-Befehlssequenzen).

Bernhard Rath, Aachen

Betr.: Artikel „Der Tatstatur auf die Sprünge ...", ST Computer 5/87
In der Mai-Ausgabe haben Sie drei Hilfsprogramme zur Belegung der Tastatur vorgestellt. Für das Programm „ALT" hatten Sie leider keine Bezugsquelle angegeben.
Bitte teilen Sie mir freundlicherweise die Bezugsquelle und den evtl. Preis mit.

Prof. Dr. E. Mehdorn, Universität Lübeck

Sehr geehrter Prof. D. Mehdorn,
Leider hat dieses Programm zur Zeit in Deutschland noch keinen Distributor. Produziert wird es von der amerikanischen Firma Michtron, ist allerdings in Europa bei der englischen Firma Microdeal zu beziehen. Näheres können wir erst nach der ATARI-Messe in Düsseldorf (18.–20.9.) sagen, da dort laut Aussage von Michtron mit deutschen Distributoren verhandelt werden soll. Die englische Bezugsadresse lautet:
Microdeal
Box 68 St. Austell
Cornwall PL25 4YB
England

Betr.: Artikel „Änderungen am Betriebssystem", ST Computer 6/87
Ich bin nicht sicher, ob Ihnen in der Juni-Ausgabe ein Fehler unterlaufen ist, oder ob es sich hier um einen Fehler meinerseits handelt. In besagtem Artikel haben Sie ein Listing abgedruckt, daß die Icons des ST auf dem Bildschirm darstellen sollte. Nachdem ich es abgetippt hatte, mußte ich feststellen, daß es zumindest auf meinem 260er nicht funktionierte. Nach einigen Überlegungen habe ich es folgendermaßen umgeschrieben:

```
A=16614022
B=0
Do
   Inc B
   Dpoke Xbios(2)+B*80,Dpeek(A)
   Dpoke (Xbios(2)+2)+B*80,Dpeek
                              (A+2)
   Add A,4
   Exit If A>=16615302
Loop
```

In dieser Version war es hundertprozentig lauffähig. War dies nun ein Druckfehler Ihrerseits, hatte es mit meinem 260er zu tun oder war es ganz einfach ein Mißverständnis meinerseits?

Urs Gulba, Grünkraut

Sehr geehrter Herr Gulba,
Sie haben Recht. Der Autor hatte das Listing leider nur auf seinem eigenen ST (1 MB) getestet und (das ist das größte Manko) nur direkt auf den Speicher zugegriffen, so daß das Listing gar nicht auf einem 260er laufen konnte. Wir möchten uns bei Ihnen im Namen unserer Leser bedanken, daß Sie dieses Listing allen ST-Besitzern zugänglich gemacht haben, indem Sie es durch den Zugriff auf die Adresse des physikalischen Bildschirms sauber programmiert haben.

Inserentenverzeichnis

In der nächsten

ST COMPUTER

lesen Sie unter anderem

STEVE

Das Programmpaket mit Extras

STEVE'S Fähigkeiten liegen in vielen Bereichen. Neben einem Grafikfähigen Textverarbeitungsprogramm enthält es eine komplette Datenbank, die einiges zu bieten hat. Im Duett sind sie unschlagbar.

Die Neuronen, die aus dem Rechner kommen

Von Neuronen und Computern

Was wäre das menschliche Gehirn ohne Neuronen? Dieser Artikel berichtet über deren Aufbau und Nachahmung im Computer. Künstlicher und natürlicher Intelligenz.

Her mit den Daten

Großer Datenbankvergleich

Inzwischen gibt es viele Datenbanken für den ST, doch nur einige tragen diesen Namen zu Recht. Welche Programme es sind und was sie leisten, zeigt dieser Bericht.

Weiche Bits für harte Chips

Die Software zum Easytizer

In diesem Heft konnten Sie sich von der Hardware unseres Digitizers überzeugen, im nächsten Heft zeigen wir, wie man ihn programmiert.

Die ST-Computer Ausgabe 11/87 erscheint am 30. 10. 1987

Impressum

ST-Computer

Verlag:
Heim Fachverlag,
Heidelberger Landstraße 194,
6100 Darmstadt 13,
Telefon (0 61 51) 5 60 57 FAX 0 61 51 / 5 56 89

Verlagsleitung:
Hans-Jörg Heim

Redaktion:
'Merlin' Computer GmbH
Uwe Bärtels (UB Chefredakteur)
Marcelo Merino (MM)
Harald Egel (HE)
Harald Schneider (HS)

Anschrift:
'Merlin' Computer GmbH
ST-Computer Redaktion
Postfach 59 69
Industriestr. 26
6236 Eschborn
Tel. 0 61 96 / 48 18 11 FAX 0 61 96 / 4 11 37

Redaktionelle Mitarbeiter:
Markus Nerding (MN), Uli Eickmann (UE), Jürgen Leonhard (JL), Jörg Wilhelm (JW), C. P. Lippert, R. Hofmann, Oliver Joppich (OJ), Stefan Höhn (SH), Chr. Schormann (CS), C. Brod, A. Suchy

Kommunikation und Presseleitung:
Claus P. Lippert

Autoren dieser Ausgabe:
Dr. Sarnow, S. Schuler, U. Litzkendorf, H. Huptach, M. Spehr, H. J. Reiss, D. Brockhaus, F. Mailhes, W. Thomas, P. Monadjemi, M. Schuhmacher

Produktion:
Klaus Schultheis (Ltg.),
Bernd Uwe Failer
Susanne Failer
Bela Kumar
Ilka Lindemann

Anzeigenverkaufsleitung:
Uwe Heim

Anzeigenpreise:
nach Preisliste Nr. 2, gültig ab 1.7.86

Vertrieb:
Hans-Jörg Heim, Uwe Heim,
Heide Schultheis

Erscheinungsweise:
11 x jährlich

Bezugspreis:
Einzelheft DM 6,–.
Jahresabonnement DM 60,– inklusive der gesetzlichen Mehrwertsteuer und den Zustellgebühren für 11 Ausgaben.
Ausland: 80,– DM inkl. Versand

Bezugsmöglichkeiten:
ATARI-Fachhändler, Zeitschriftenhandel, Kauf- und Warenhäuser oder direkt beim Verlag unter obiger Adresse.

Druck:
Ferling Druck Darmstadt

Manuskripteinsendungen:
Programmlistings, Bauanleitungen und Manuskripte werden von der Redaktion gerne angenommen. Sie müssen frei von Rechten Dritter sein. Mit ihrer Einsendung gibt der Verfasser die Zustimmung zum Abdruck und der Vervielfältigung auf Datenträgern im Heim Verlag. Honorare nach Vereinbarung. Für unverlangt eingesandte Manuskripte wird keine Haftung übernommen.

Titelseite: Fabian & Mayer
Foto: Rainer Spirandelli